2[nd] Annual GSFC-JPL Quality Mission Software Workshop

Goddard Space Flight Center
Jet Propulsion Laboratory

Session 2: Future Infrastructure and Technology for Software Development

San Diego, California
May 16-18, 2000

AGENDA

Session 2: Future Infrastructure and Technology for Software Development

Day 1
Tuesday – May 16, 2000

Future Infrastructure and Technology for Software Development

1:00 pm	An Overview of State-of-the-Practice/Art for Middleware	N. Lamarra
1:30 pm	Middleware Tools (Part 1)	D. Crichton
2:00 pm	Middleware Tools (Part 2)	T. Ames
2:30 pm	ISE Mars Application	E. DeJong
3:00 pm	Break	
3:15 pm	Automated Software Verification	K. Havelund
3:45 pm	Working Session on future Collaboration Opportunities	All
5:00 pm	End of First Day	

AGENDA

Session 2: Future Infrastructure and Technology for Software Development

Day 1
Tuesday – May 16, 2000

Future Infrastructure and Technology for Software Development

1:00 pm	An Overview of State-of-the-Practice/Art for Middleware	N. Lamarra
1:30 pm	Middleware Tools (Part 1)	D. Crichton
2:00 pm	Middleware Tools (Part 2)	T. Ames
2:30 pm	ISE Mars Application	E. DeJong
3:00 pm	Break	
3:15 pm	Automated Software Verification	K. Havelund
3:45 pm	Working Session on future Collaboration Opportunities	All
5:00 pm	End of First Day	

2nd Annual GSFC-JPL Quality Mission Software Workshop

Future Infrastructure & Technology for Software Development

Overview of SOP/SOA for Middleware

JPL

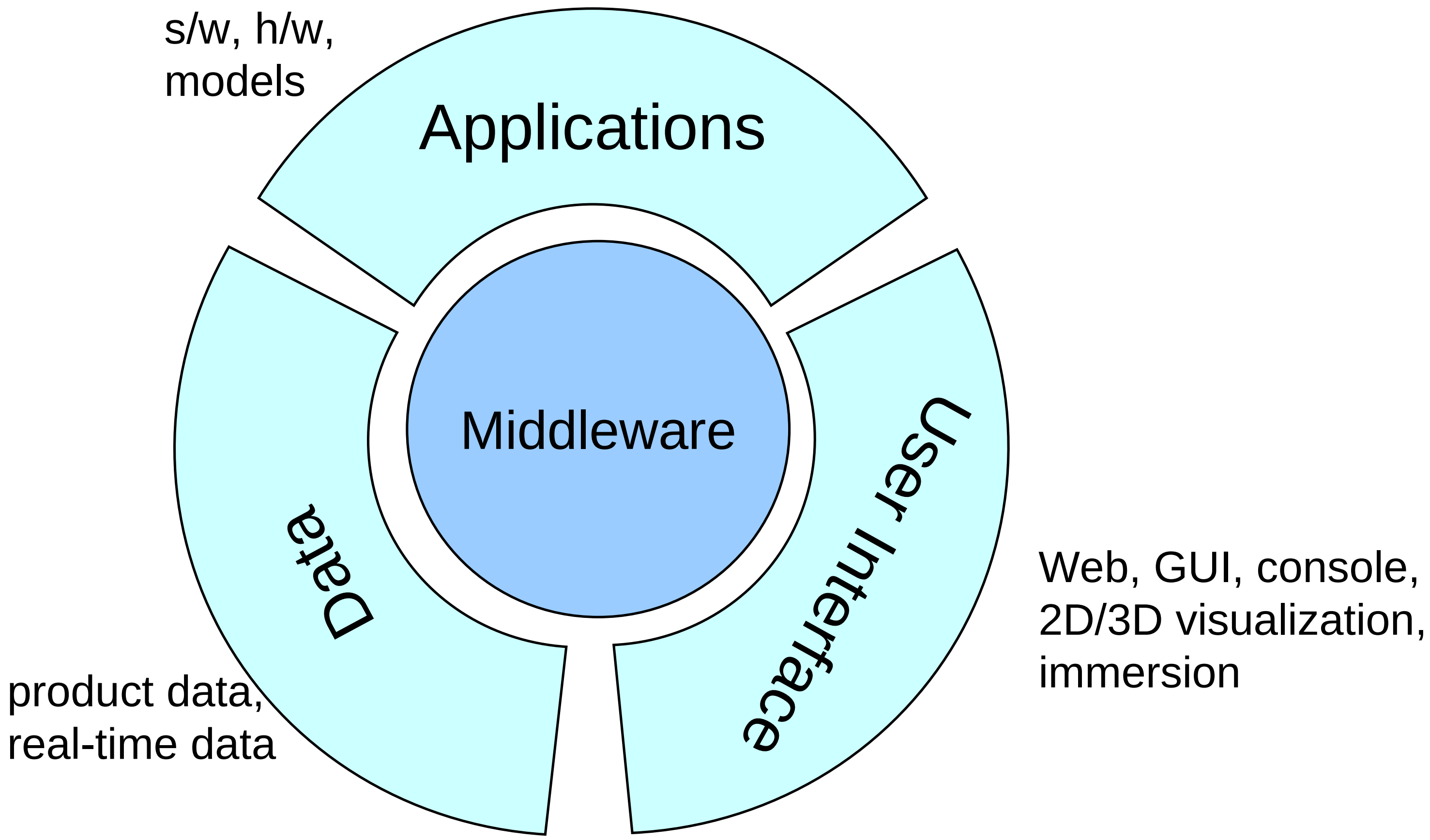
Norman Lamarra, Ph.D.

May 16, 2000

Overview

- Simplest Architecture Definition
- Middleware Review
  - History, Technologies, Standards
- Middleware Techniques for Improving Software Quality
- Example Applications (SOP)
- State Of the Art (SOA)
- Crystal Gazing
- Resources for Further Information

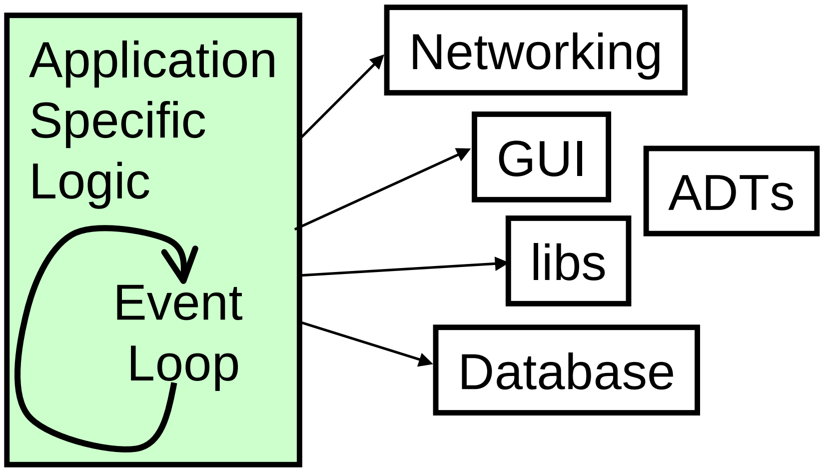
Simplest Architectural Definition



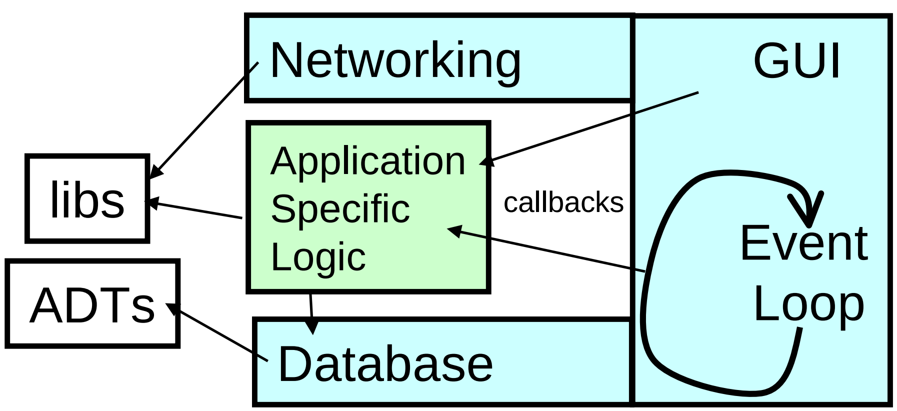
Middleware Review

- History
  - Pre-web: IDE (dev), client/server (apps), NOS (services)
  - Post-web: n-tier c/s, thin/thick, frameworks
- Technologies
  - Communications: RPC, MOM, Web, Directory
  - Applications: DOC, Intelligent Agents, Systems Mgt, Languages
  - Data: Database, PDM, Transactions
  - Architecture: Open Reference Model, HLA, TOGAF
- Standards
  - O/S: POSIX, Unix, Windows
  - Communications: TCP/IP, OSI, DCE/ONC, HLA/RTI, HTTP
  - DOC: CORBA/COSS, DCOM/COM+, Java RMI
  - Data access: O/JDBC, EJB persistence, STEP, HTML, XML
  - Security: SSL, SSH, SHTTP, IPSec, PKI, CDSA, BioAPI, etc.
  - Directory: X.500, LDAP
  - Systems Management: CIM, SNMP, CMIS/CMIP
  - GUI: X11, CDE, Java-Swing

Middleware Techniques for Improving Software Quality and Productivity



Class Library Architecture



Framework Architecture

- Components
  - pluggable ADTs
- Frameworks
  - reusable, semi-complete apps
- Patterns
  - problem/solution/context
- Architecture
  - families of related patterns & components

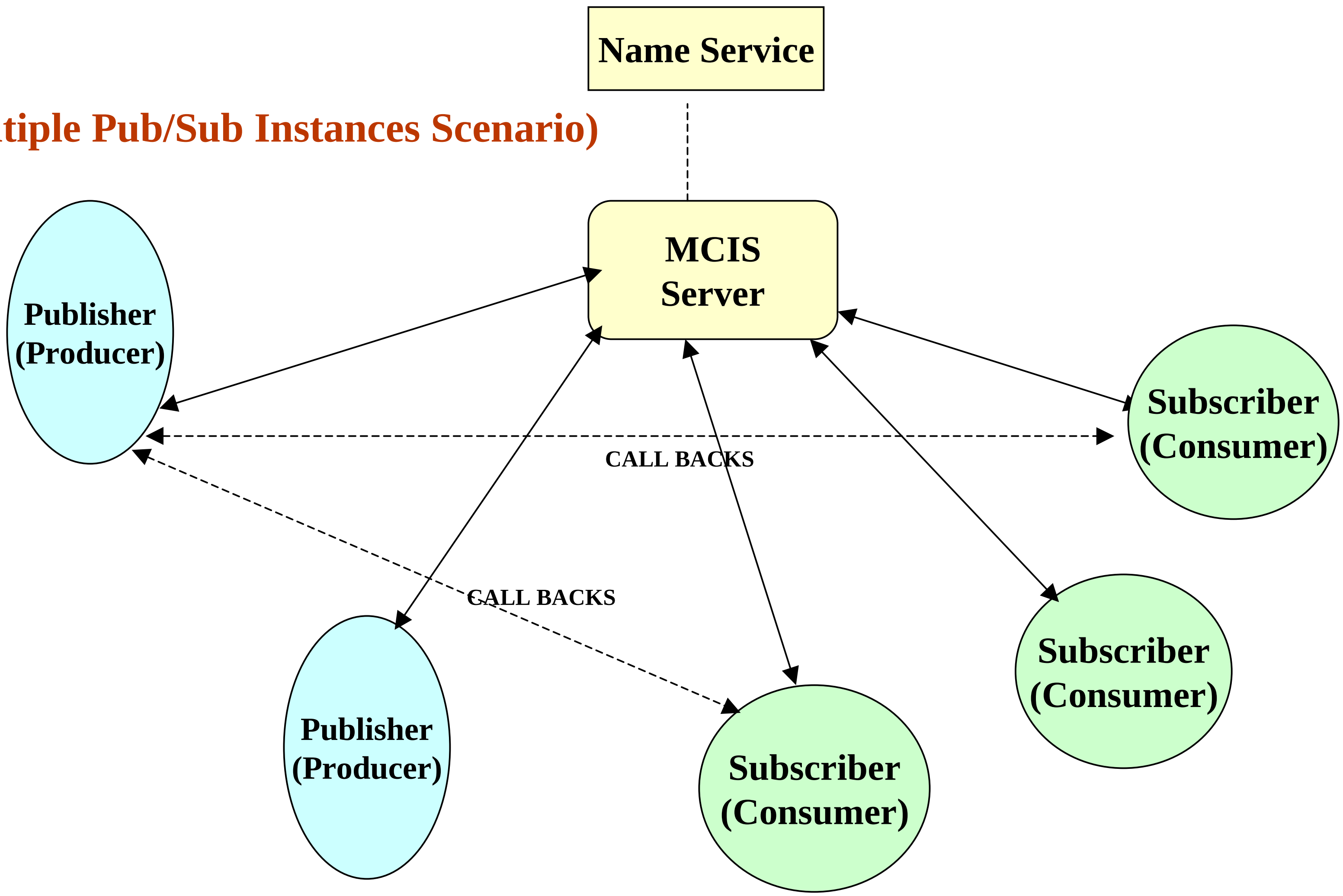
From Douglas Schmidt (schmidt@uci.edu)
www.cs.wustl.edu/~schmidt

Example Applications (SOP)

- JPL Common Software (next 3 charts)
  - Distributed Computing & System Engineering Group, Sec 369
- Military Simulation
  - AFRL's CEE (cross-domain technology development, design, trade study, test, training)
    - Enterprise Common Object Model, Smart Enterprise Model
    - IPPM + Workflow
  - Joint Synthetic Battlespace (n/w of VE)
- E-commerce
  - Integrated business processes (customer i/f, supply chain)

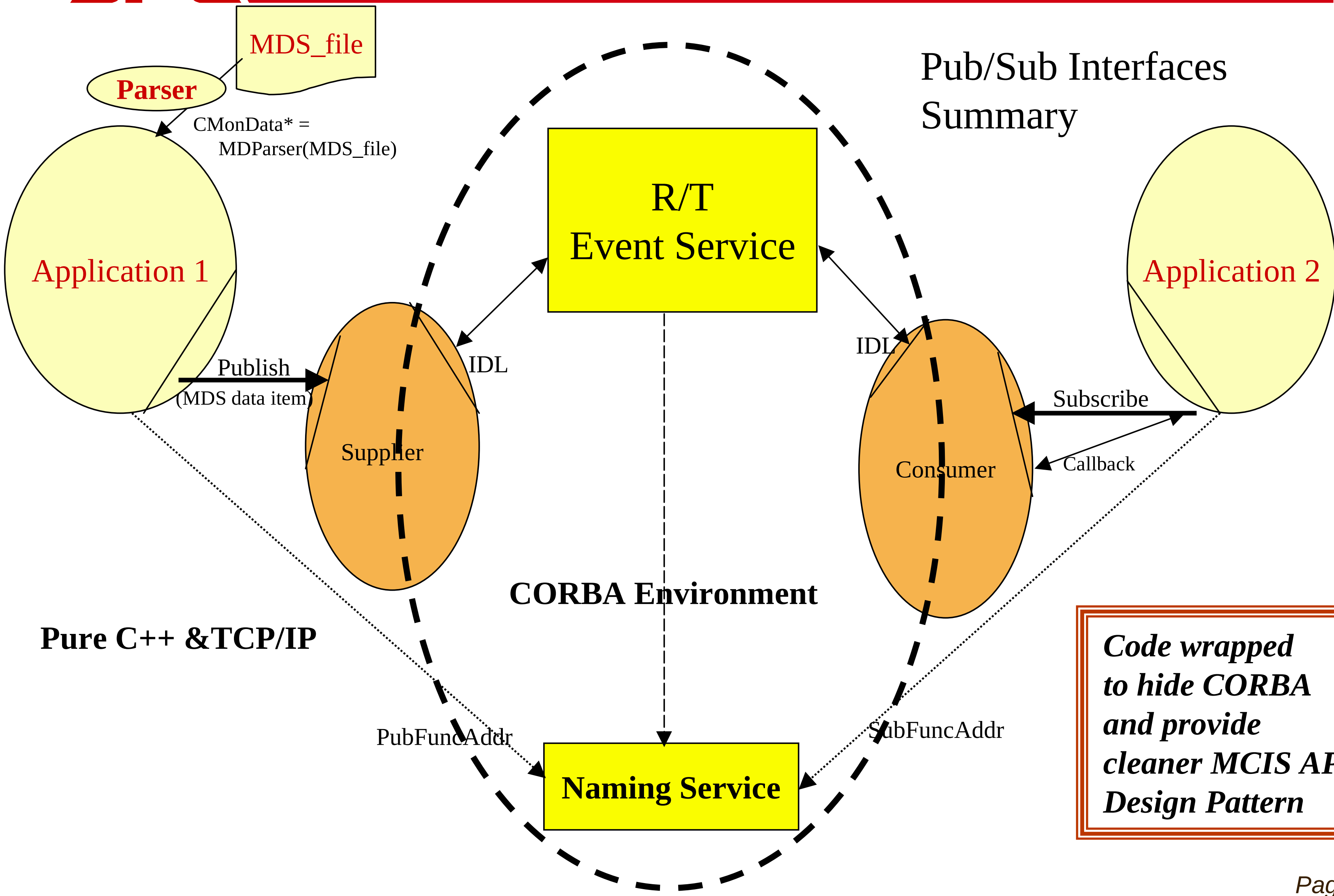
JPL Example:
DSN MCIS Pub/Sub Architecture

(Multiple Pub/Sub Instances Scenario)



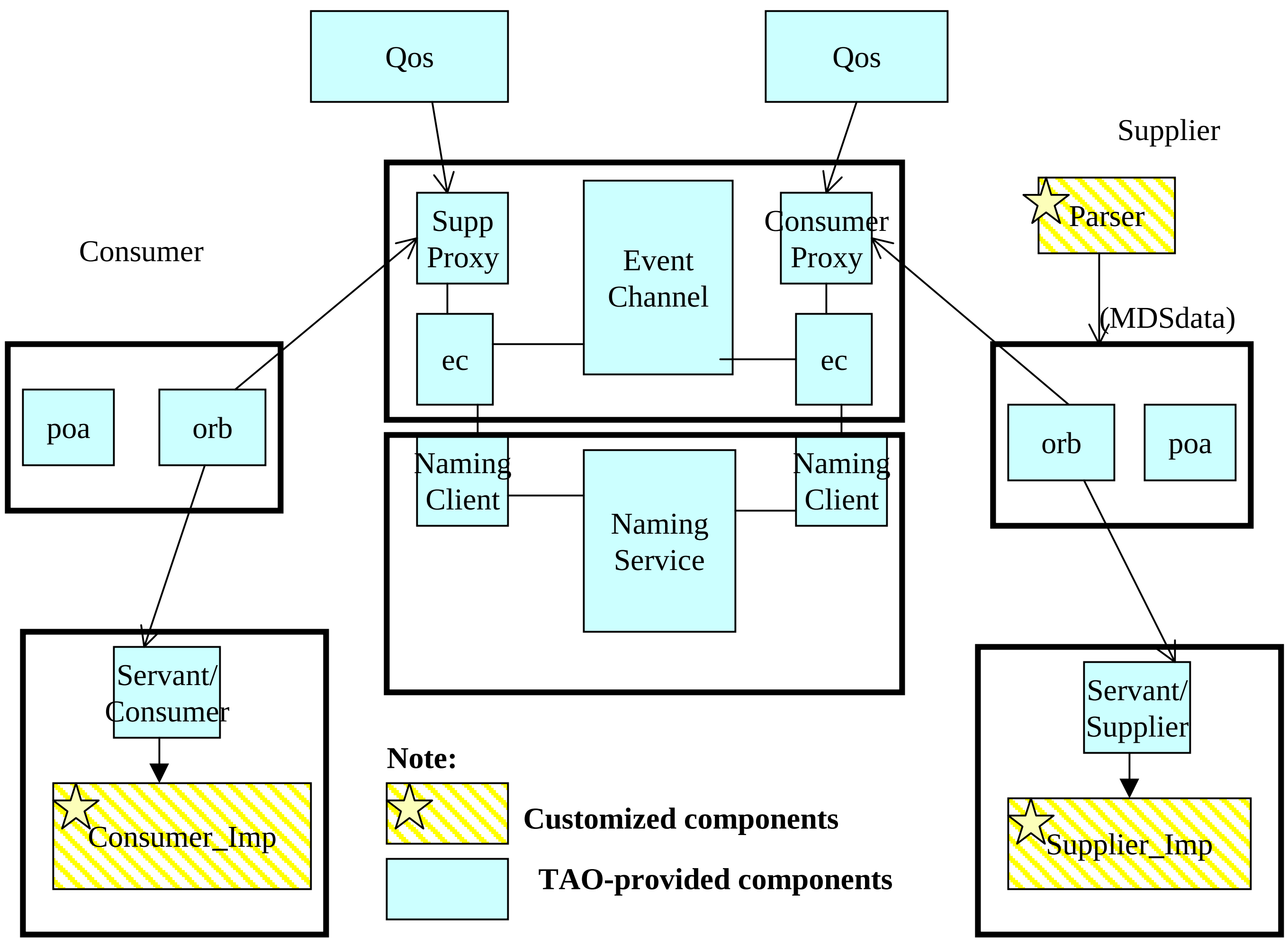
MCIS - Monitor & Control Infrastructure Services

JPL ReUse: Pub/Sub using ACE-TAO

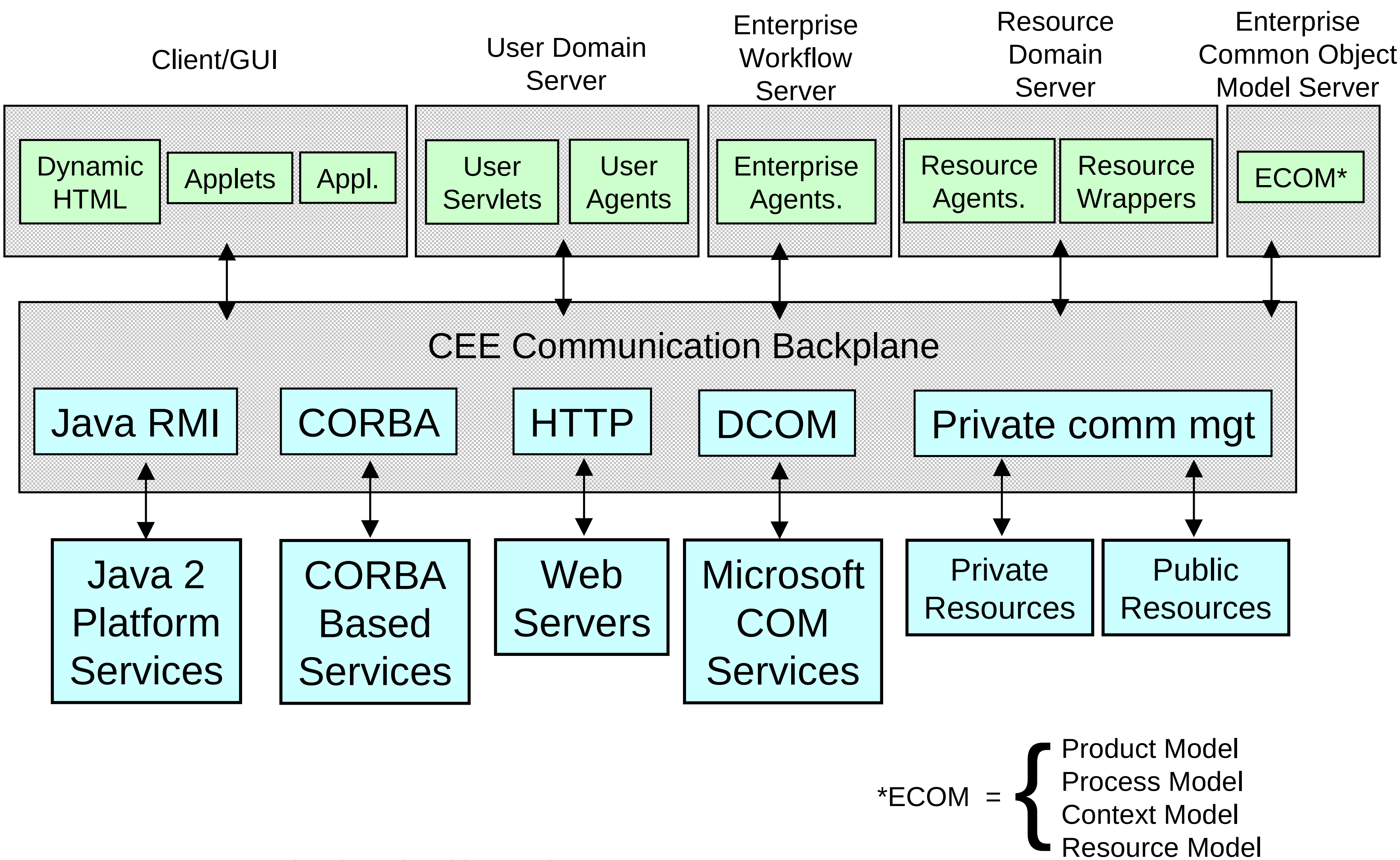


Code wrapped to hide CORBA and provide cleaner MCIS API - Design Pattern

JPL: CORBA Components used for MCIS Implementation

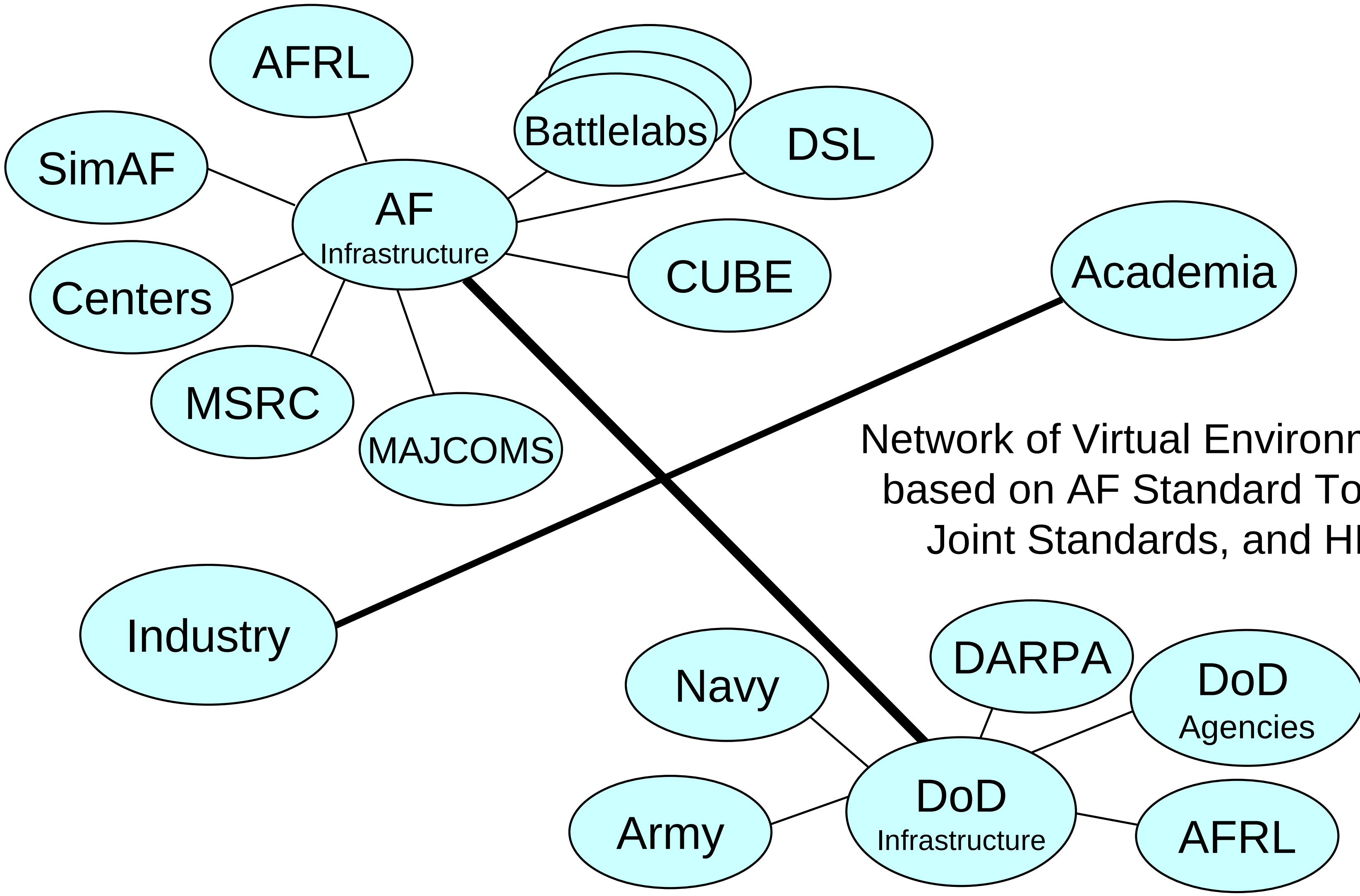


AFRL CEE Standards-Based Architecture



Vance Saunders (saunders@ball.com)

DOD Joint Synthetic Battlespace



Network of Virtual Environments, based on AF Standard Toolkit, Joint Standards, and HLA

Bill McQuay (AFRL)

State Of the Art

- Communications
  - Portable frameworks, e.g., ACE (performance, reusability)
  - Adaptive Networks, VPN, DEN, QoS, Fault-tolerance
- Distributed Object Computing
  - CORBA 2.3 (NS, ES, LCS, OTS, GSS, IDL, IOP)
    - TAO (performance, configurability)
  - CCM (server-side apps)
  - EJB 1.0 (session vs. entity beans, stateless vs. stateful)
  - COM+
- Enterprise Application Integration (EAI)
  - MQ, Forte, WebLogic, Netscape, Active, Template, Microsoft, etc.
- Architecture-Based Systems: TOGAF; ATAM, PLP (CMU)

Crystal Gazing

- “Commodity” middleware
  - Heterogeneous interoperable components (e.g., CCM)
  - Customizable frameworks (specialized, e.g., reliable)
  - Reusable architectures
- “Today’s EAI will be tomorrow’s IT architecture”
  - better information value
  - evolving processes to optimize workflow & ERM
- Broadband WAN
  - Location independence (intelligent n/w; wireless)
- Virtual Environments
  - Remote interactivity (Marsnet, etc.)
- Quantum computing
  - communications, coding

Resources for Further Information

- Conference:
  - www.research.ibm.com/Middleware2000/
  - www.comp.lancs.ac.uk/computing/RM2000/
- Open Systems Standards: www.opengroup.org
- CORBA: www.corba.org, www.omega.org
- ACE/TAO: www.cs.wustl.edu/~schmidt
- STEP: pdesinc.scra.org/
- COM/COM+: www.microsoft.com/com
- Java: java.sun.com/docs/books/tutorial/
- XML: www.xml.org, www.xml.com

2nd Annual GSFC-JPL Quality Mission Software Workshop

Middleware Tools (Part I): Support for a Data Architecture

JPL

OODT OBJECT ORIENTED DATA TECHNOLOGY

ENTERPRISE DATA ARCHITECTURE

Dan Crichton

May 16, 2000

Introduction

- The Object Oriented Data Technology (OODT) task which is part of Interactive Analysis Environments (IAE) program is funded by NASA’s Science Information Systems Program.

- Research middleware solutions that provide common services for managing and providing access to science data including
  - Archive systems
  - Search and retrieval systems
  - Methods for interoperability and correlation
  - Support for thin clients

Goddard Space Flight Center JPL

Problem Statement

- Currently, data systems at NASA
  - Are difficult to access (no standard interfaces)
  - Are Geographically distributed
  - Have no standard language or protocol for interchange (no EDI)
  - Have no system for registration of data products
  - Have little or no interoperability

- No Agency-wide Data Architecture

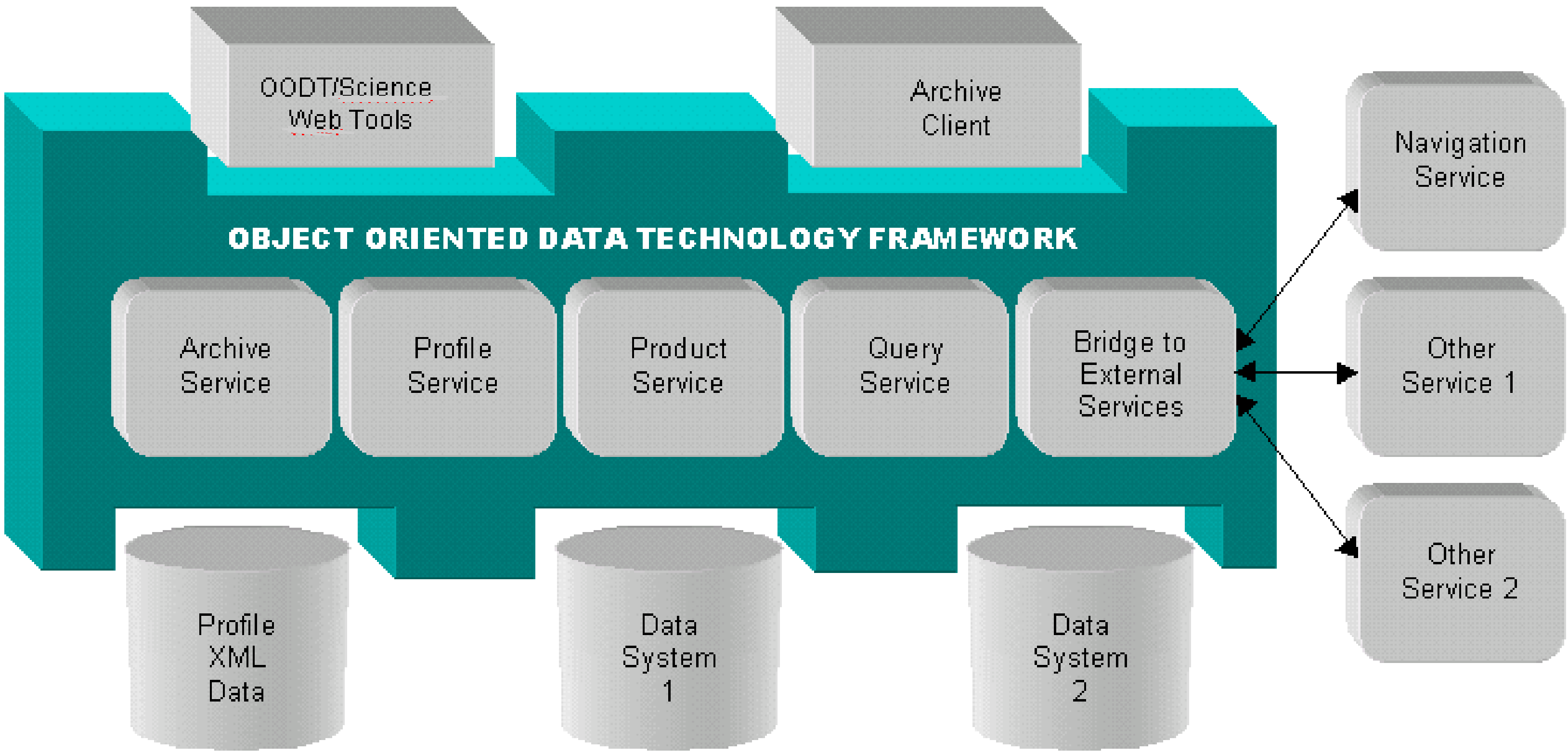
Problem Example

- Space scientists can not easily locate or use data across the hundreds if not thousands of autonomous, heterogeneous, and distributed data systems currently in the Space Science community.
  - Heterogeneous Systems
    - Data Management - RDBMS, ODBMS, Home Ggrown DBMS, Binary Files
    - Platforms - UNIX, LINUX, WIN3.x/9x/NT, Mac, VMS, …
    - Interfaces - Web, Windows, Command Line
    - Data Formats - HDF, CDF, NetCDF, PDS, FITS, VICR, ASCII, ...
    - Data Volume - KiloBytes to TeraBytes
  - Heterogeneous Disciplines
    - Moving targets and stationary targets
    - Multiple coordinate systems
    - Multiple data object types (images, cubes, time series, spectrum, tables, binary, document)
    - Multiple interpretations of single object types
    - Multiple software solutions to same problem.
    - Incompatible and/or missing metadata

Proposed Solution

- *Encapsulate individual data systems.*
  - *Hide uniqueness by introducing a middleware layer*
- *Build an EDI capability through middleware services*
  - *Provides a language for communication between client and server systems*
- *Focus on a metadata infrastructure for interoperability*
  - *Create a standard language for communication*
- *Use industry standard technologies including Java, CORBA and XML*
- *Provide a core set of middleware components*
  - *Archive Service*
    - *An extensible framework for building science archive systems*
  - *Query Service*
    - *A server for managing concurrent, distributed queries*
  - *Profile Service*
    - *A server for managing content information*
  - *Product Service*
    - *A mapping service for translating product information from proprietary data systems into an exchange language*

Middleware Component Architecture



Middleware Design Decisions

- Driving Requirements for a middleware solution:
  - ➤ Accept and introduce a solution that addresses NASA's heterogeneous data system culture
  - ➤ Select technologies that are cross-platform, scalable, and extensible
- Develop CORBA servers written in Java
  - ➤ Provides a number of basic services for distributed object management
    - Security, Naming, Trading
  - ➤ Provides support for multi-language clients
- Develop Interfaces between CORBA services using XML
  - ➤ Flexible (Allows interfaces to evolve)
  - ➤ Self-describing
  - ➤ Combines data and metadata
  - ➤ OMG and W3C cooperation

CORBA vs. XML

- CORBA method

```
module jpl {
   module user {
     interface UserManager {
        User findUser(string
                          name);
     };
     interface User {
       String getName();
     };
   };
 };
```

- XML over CORBA/IIOP

```
module jpl {
   module user {
     interface UserManager {
       string do(string xml);
     };
   };
 };
```

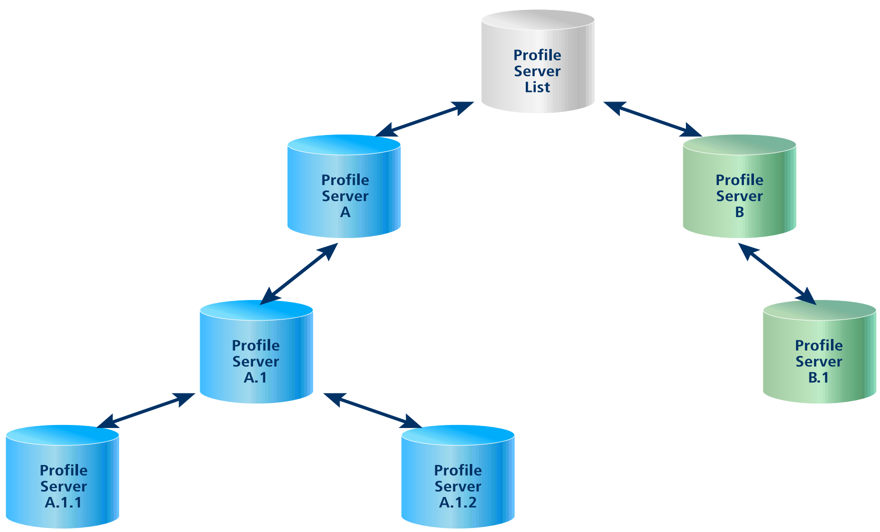
```
<transaction>
   <findUser>
     <user>
      <surname>Doe</surname>
     </user>
   </findUser>
</transaction>
```

Why XML for OODT?

- XML doesn't provide a "silver bullet", but it does allow us to refocus the problem on metadata
  - Metadata is a key to interoperability
- XML is language neutral
- Allows the designer to separate the data and the transport (re: CORBA vs XML-over-CORBA)
  - Transport mechanism and data are not tied together
    - Could be XML/HTTP
    - Simpler deployments
    - Simpler interfaces
    - Allows technologies to grow and change independently
- Real value of XML is the content

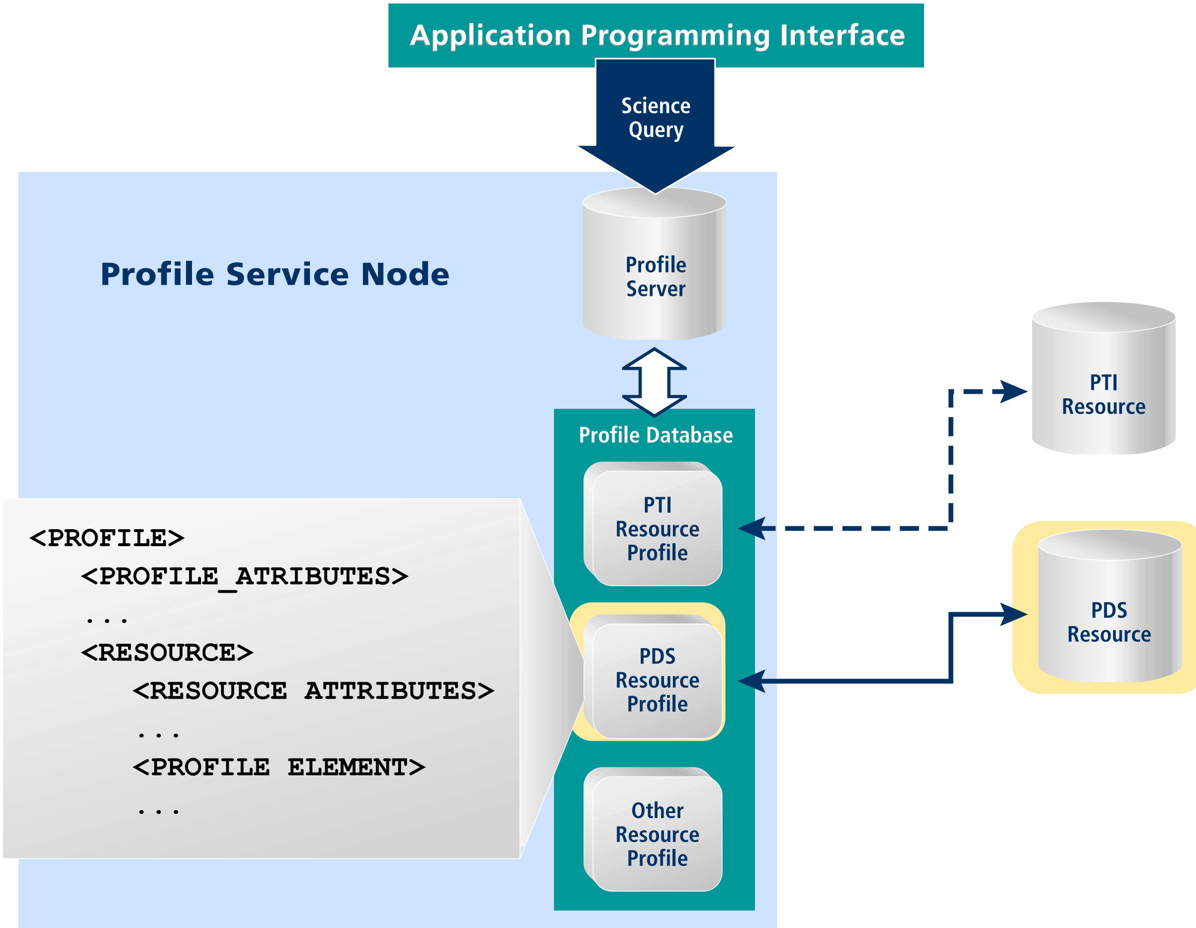
An OODT Profile

- A profile is a set of resource definitions implemented in XML for data products residing in one or more distributed systems
- Profile servers are CORBA servers that manage XML profile definitions
- Profile servers communicate via XML-over-CORBA
- Developed Java classes that map XML profiles to a Java object



Profile Distributed Node Architecture

Profile Server Architecture



Contents of a Profile

- A profile contains
  - Profile attributes - id, title, desc, type, data_dictionary_id, …
  - Resource attributes - id, title, discipline, location_id, …
  - Profile elements
    - name, syntax, unit, instance, meaning, alias, …
    - encodes all domain attributes and their values specific to this resource
- Based on Dublin Core Standard
- Implemented as a generic XML DTD that allows us to define information about any data products
  - One DTD to describe any science resource

Profile DTD

```
<!ELEMENT PROFILES
   (PROFILE+)>

<!ELEMENT PROFILE
   (PROFILE_ATTRIBUTES,
     RESOURCE)>

  <!ELEMENT PROFILE_ATTRIBUTES
     (ID, TITLE*, DESC*, TYPE*,
      STATUS_ID*, SECURITY_TYPE*,
      PARENT_ID*, CHILD_ID*,
      REVISION_NOTE*,
      DATA_DICTIONARY_ID*)>

<!ELEMENT RESOURCE
     (RESOURCE_ATTRIBUTES,
      PROFILE_ELEMENT*)>

<!ELEMENT RESOURCE_ATTRIBUTES
      (RESOURCE_ID,
       RESOURCE_TITLE,
       RESOURCE_DISCIPLINE,
       RESOURCE_AGGREGATION,
       RESOURCE_CLASS,
       RESOURCE_LOCATION_ID,
       RESULT_MIME_TYPE)>

    <!ELEMENT PROFILE_ELEMENT
      (ELEMENT_NAME,
       ELEMENT_MEANING*,
       ELEMENT_ALIAS*,
       VALUE_SYNTAX*,
       VALUE_UNIT*,
        (VALUE_INSTANCE |
         (MINIMUM_VALUE,
          MAXIMUM_VALUE))*)>
```

Profile Example

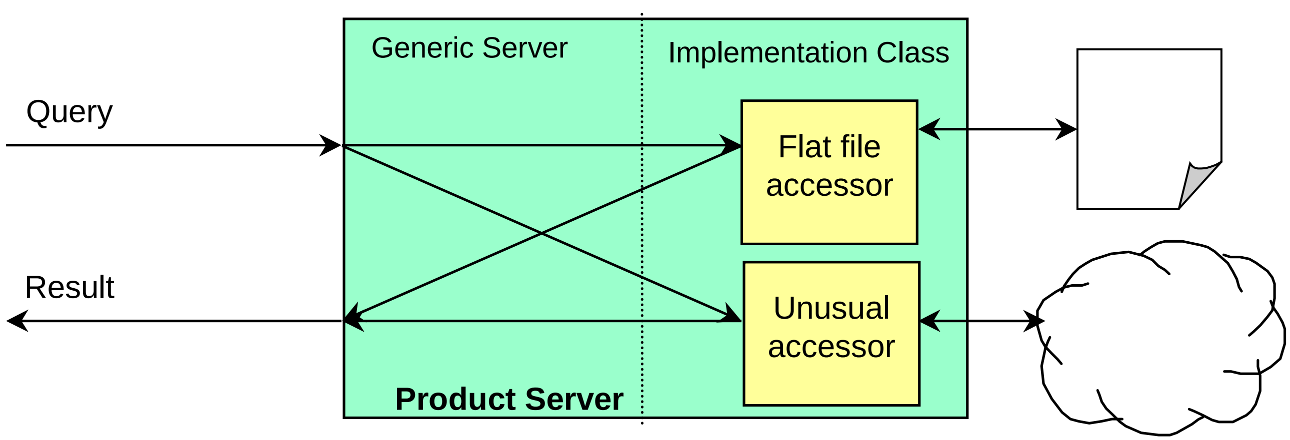
```
   ...
<PROFILE_ELEMENT>
    <ELEMENT_NAME> DATA_OBJECT_TYPE </ELEMENT_NAME>
    <ELEMENT_MEANING> The data_object_type element provides the type ...
    <VALUE_SYNTAX> ENUMERATION </VALUE_SYNTAX>
    <VALUE_UNIT> N/A </VALUE_UNIT>
    <VALUE_INSTANCE> IMAGE </VALUE_INSTANCE>
  </PROFILE_ELEMENT>
  <PROFILE_ELEMENT>
    <ELEMENT_NAME> DATA_SET_NAME </ELEMENT_NAME>
    <ELEMENT_MEANING> The data_set_name element identifies a PDS data set.  -- example ...
    <VALUE_SYNTAX> ENUMERATION </VALUE_SYNTAX>
    <VALUE_UNIT> N/A </VALUE_UNIT>
    <VALUE_INSTANCE> VO1/VO2 MARS VISUAL IMAGING SUBSYSTEM DIGITAL ...
    <VALUE_INSTANCE> VO2 MARS RADIO SCIENCE SUBSYSTEM RESAMPLED LOS ...
  </PROFILE_ELEMENT>
  <PROFILE_ELEMENT>
    <ELEMENT_NAME> TARGET_NAME </ELEMENT_NAME>
    <ELEMENT_MEANING> The target_name element provides the names of the targets ...
    <ELEMENT_ALIAS> ADS.OBJECT_ID </ELEMENT_ALIAS>
    <VALUE_SYNTAX> ENUMERATION </VALUE_SYNTAX>
    <VALUE_UNIT> N/A </VALUE_UNIT>
    <VALUE_INSTANCE> IDA </VALUE_INSTANCE>
    <VALUE_INSTANCE> JUPITER </VALUE_INSTANCE>
  </PROFILE_ELEMENT>
 </RESOURCE>
```

Electronic Data Interchange

- Distributed servers communicate using an XML query language for data interchange
  - Enforced by a Query Document Type Definition (DTD)
  - Allows for complex queries to be specified
    - Products to locate
    - Formats to convert to/from
    - Inclusion/exclusion List
    - Associates query and results in the same structure

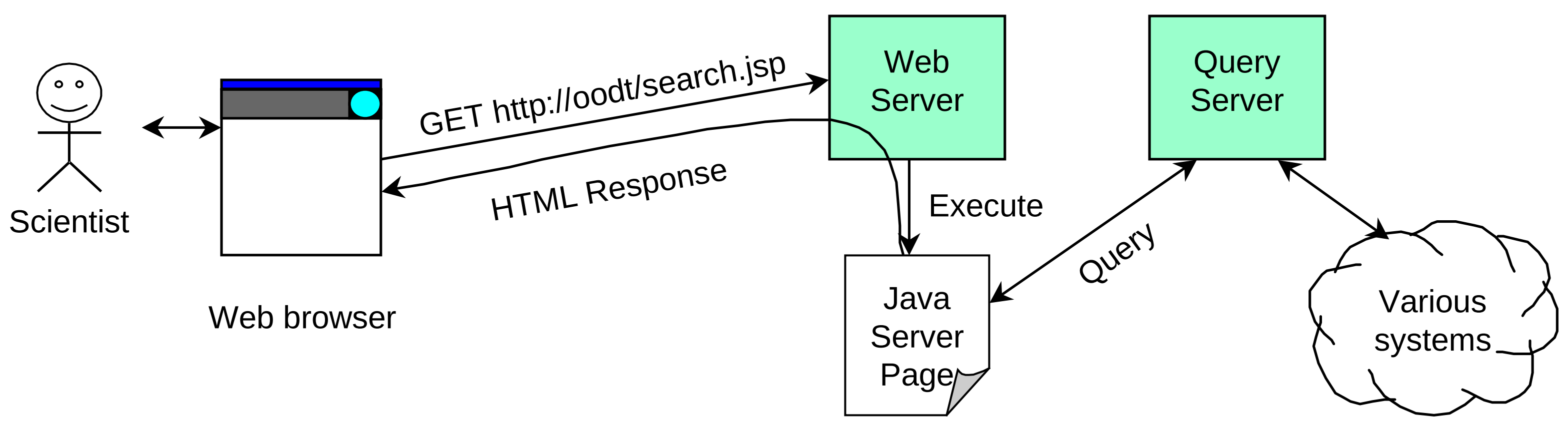
Data Transformations

- Key issue is that there is no single data model. Data types, formats, etc are all different
- The OODT architecture provides a “mapping” server or product server that translates products from their native format to a common exchange language based on the XML Query DTD
- Mapping Server is a Java-based Query Server that dynamically loads mapping objects that map queries to results



Presentation Layer

- Used Java Server Pages (JSP)
  - Part of Sun's Java 2.0 Enterprise Edition
  - Ties HTML and Java together in the Web Server
    - Java classes are accessible
  - Client is dynamic HTML (no applets)
  - Another example of middleware
    - Clients are thin (require just a web browser)



XML and XSL

- XSL (Extensible Stylesheet Language) allows XML to be rendered in HTML by browsers that support XSL
- Transformation via XSL can happen on the web server, too
  - Solves problem where browser lacks support for XSL
  - Server pushes HTML

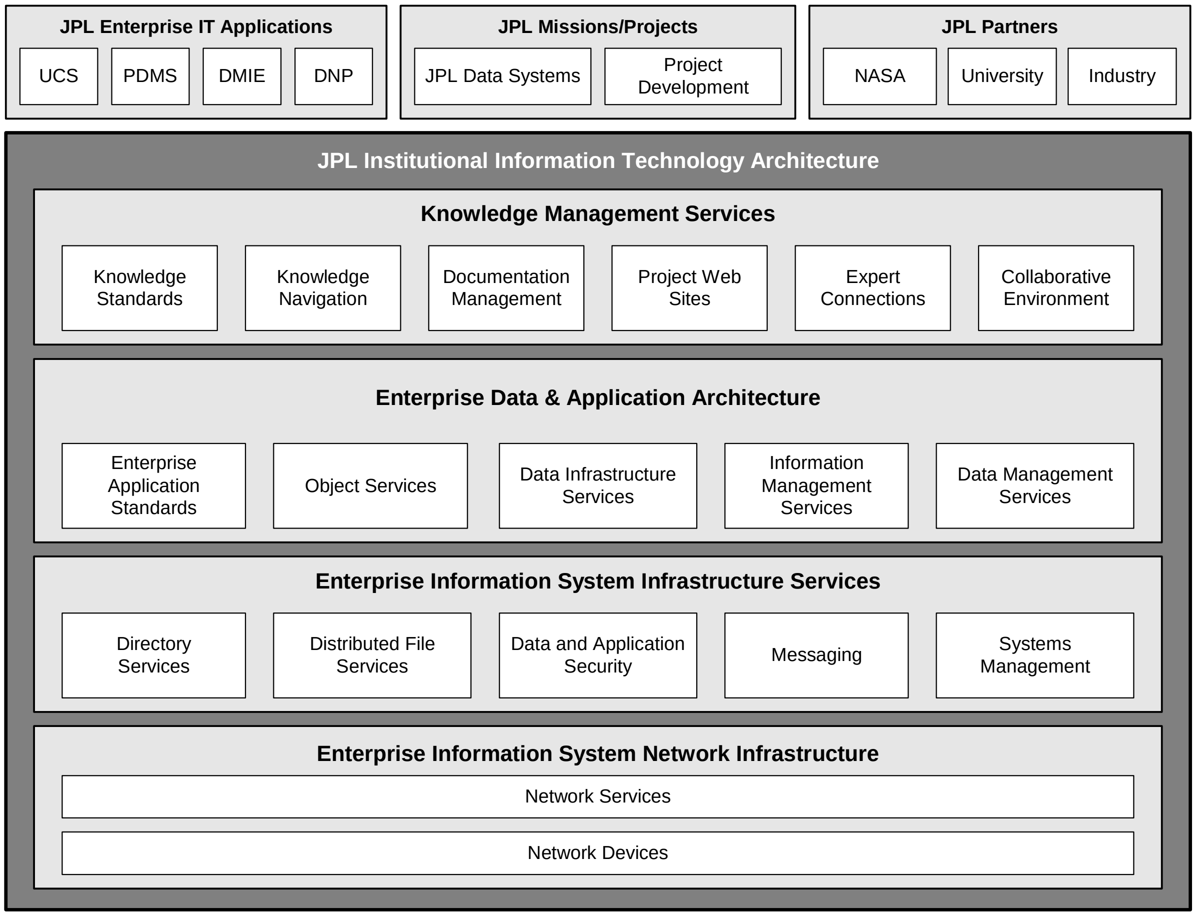
Wrapping up the OODT Framework

- The components of the framework address data management and interoperability as follows:
  - Archive Server - Extensible server driven by metadata configurations. Data storage manager.
  - Profile Server - Metadata manager
  - Product Server - Translation server
  - Query Server - Search and retrieval query manager
- Current partners/Data providers
  - Planetary Data System
  - Palomar Testbed Interferometer
  - SeaWinds

Other Examples of Middleware

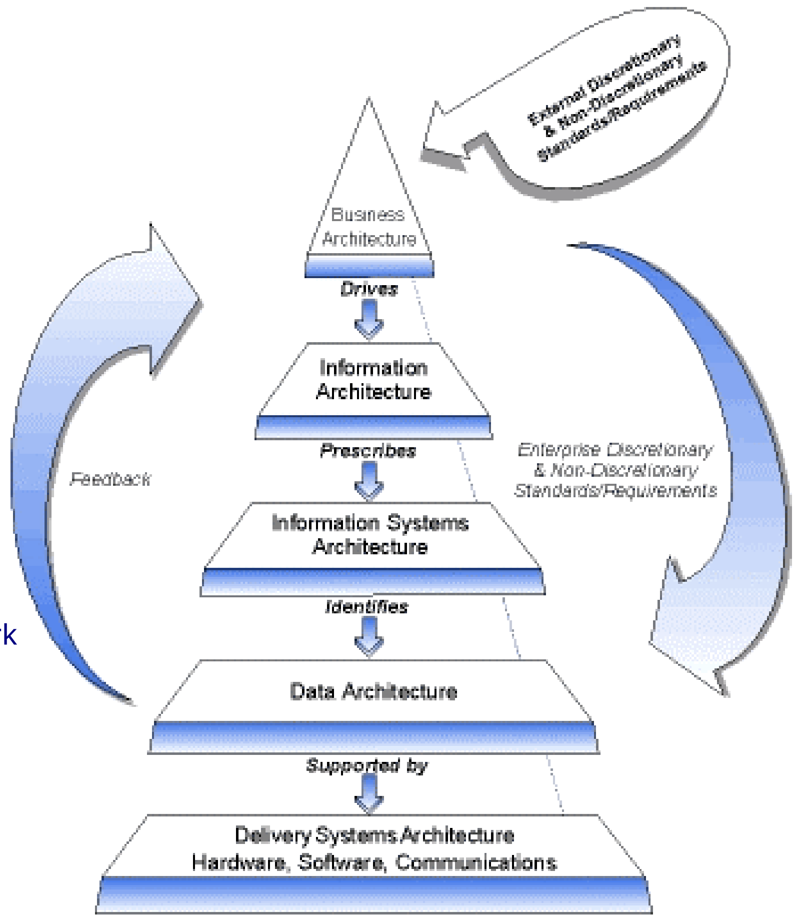
- The Enterprise Data Architecture
  - The Institutional Computing and Information Services (ICIS) is building a data architecture that ties together the institutional applications and institutional services
    - Provides basic set of core middleware services
      - Metadata Management
      - Object Management
      - Data and Information Management
      - Data Exchange
    - Abstracts client applications away from institutional services
    - Provides a facility for interoperability with other centers
  - Institutionally supported middleware that begins to create an *information architecture (E.g. transform data into information)*

JPL IT Architecture



NIST I.T. Architecture for Federal Gov't

Example Architecture at the Federal Level



Chief Information Officers Council

Federal Enterprise Architecture Framework

Version 1.1

September 1999

Conclusion and Future Work

- XML for large datasets
  - Investigate use of OODT XML profiles for accessing PDS data
    - XML is descriptive, but wordy. What's the feasibility.
  - Store Java profiles in a DBMS or XML server
- Investigate XML/HTTP
- Investigate NISO Z39.50 standard for information retrieval
- Work to insert developed technology into the JPL and NASA IT architectures
  - Focus on developing standards for interoperability and communication
  - Focus on building an infrastructure for EDI
- Support technology transfer
  - NIH

More Information

- "Science Search and Retrieval using XML" by OODT Team. Presented at Second National Conference on Scientific and Technical Data, National Academy of Sciences, Washington D.C.

  http://oodt.jpl.nasa.gov/doc/papers/codata/paper.pdf
- Dublin Core

  http://purl.oclc.org/dc
- Extensible Markup Language

  http://www.w3c.org/XML
- ISO/IEC 11179: Specification and Standardization of Data Elements
- Object Management Group (CORBA and UML standards)

  http://www.omg.org
- Federal CIO Statement on Metadata

  http://www.cio.gov/docs/metadata.htm
- National Information Standards Organization Z39.50 Information Retrieval Protocol

  http://www.niso.org/z3950.html

Future Infrastructure and Technology for Software Development

Middleware Tools (Part 2): Using XML and Java for Telescope and Instrumentation Control

Advanced Architectures & Automation Branch
Information Systems Center
Goddard Space Flight Center

Troy Ames

May 16, 2000

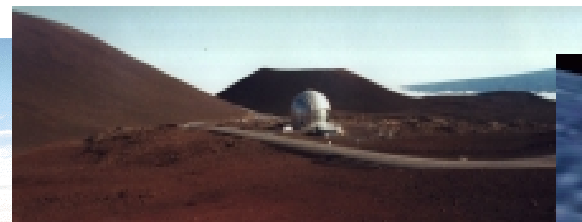
Agenda

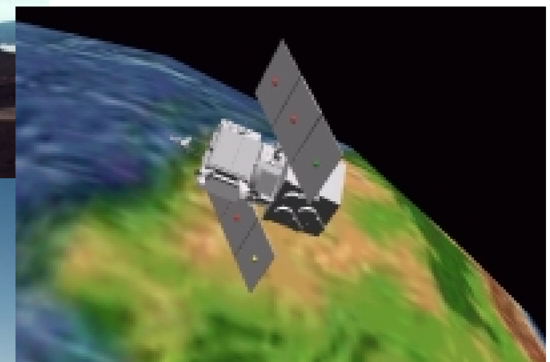
- Background and Goals
- XML and Java™ Technology
- Instrument Markup Language (IML)
- Architecture
- Major Developments
- The Vision
- Next Steps

Instrument Remote Control Project

Adaptive framework from science investigator to instrument that provides remote configuration, control, monitoring, and data analysis.

Airborne





Space-based

Ground

IRC Project Goals

- Extensible Framework for Remote Control of (Astronomical or other) Instruments
  - promote reuse by design
  - easier to develop, modify, maintain, extend
  - capability to integrate new technology
- Clearly Defined Interface
  - hardware engineers describe instruments naturally
  - software uses instrument description automatically
- Support for Iterative Development
- Platform independent solution
- Reduction of development costs of future ground, airborne, and space-based instrumentation projects.

Java™ Technology

- Object Oriented
- Reusable components
- Rich set of APIs
  - Networking
  - Graphics
  - GUI
  - Security
- Platform independent
  - Java™ 2 Platform, Enterprise Edition
  - Java™ 2 Platform, Standard Edition
  - Java™ 2 Platform, Micro Edition
  - EmbeddedJava™
  - PersonalJava™, Java Card™, JavaPhone™, Java TV™, JavaOS™

XML - Extensible Markup Language

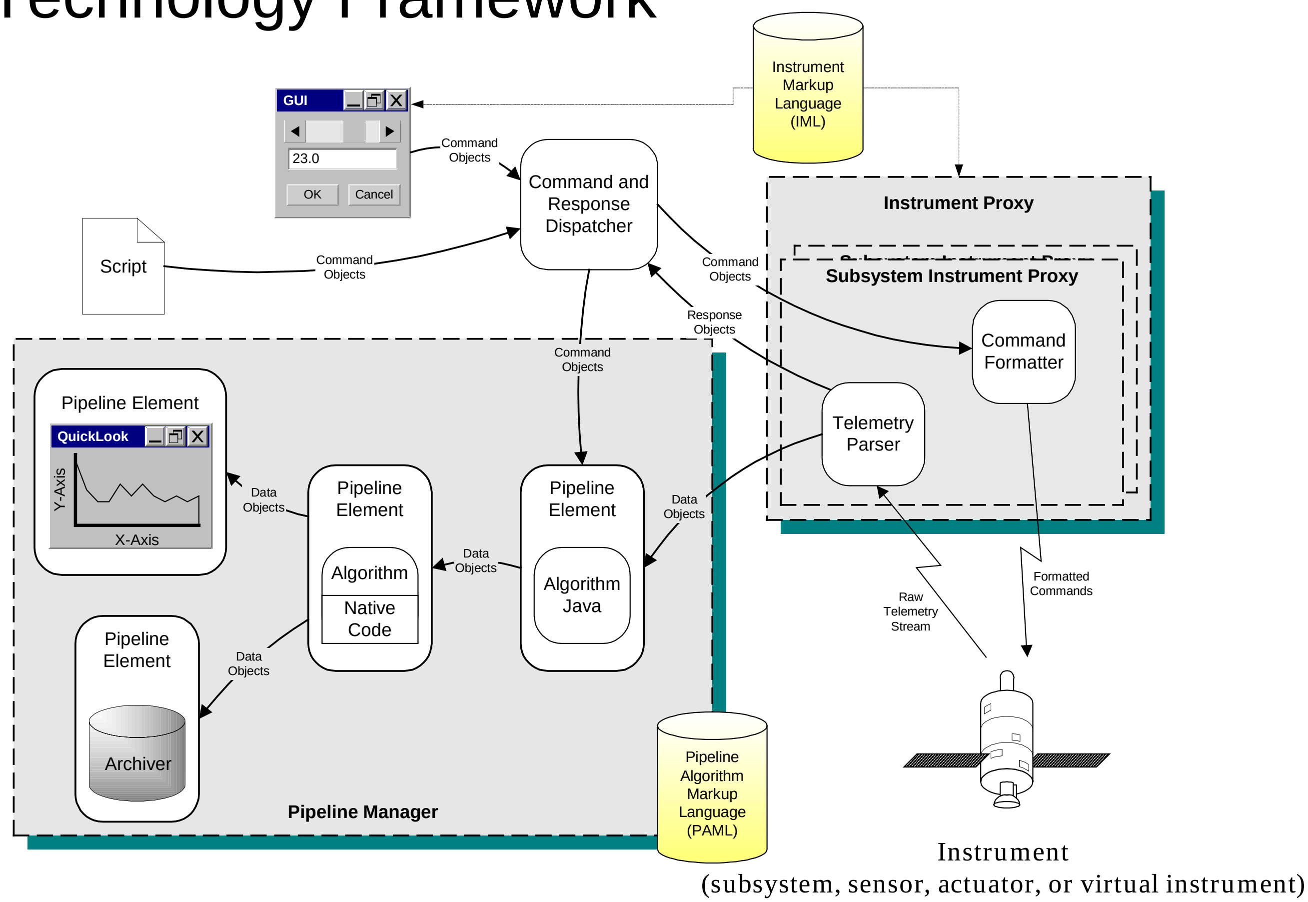
- Metalanguage -- a language for describing other languages
  - Can construct specific dialects
  - Extensible vocabularies
- Structured, hierarchical data
  - Human readable
  - Machine-understandable
  - Independent of representation
- Platform independent
- Tools (e.g., Parsers & Editors)

Java™ Technology and XML

- Designed for distributed systems
  - Java Technology designed to be delivered over networks to different devices
  - XML designed to be simplified SGML for the Web
- Grew up with Internet community
  - Both are new technologies
  - Many Java technology based XML parsers
- Complementary features
  - Platform independent
  - Designed to be easily shared
  - Support Unicode

Instrument Remote Control Architecture

- XML-based Instrument and Pipeline Description
- Java Technology Framework



Instrument Markup Language

- IML is an XML dialect to describe instruments
  - Logical command set
  - Command arguments and units
  - Argument data types and valid values/ranges
  - Command formats
  - Logical data streams
  - Data fields (telemetry)
  - Data field data types
  - Data formats
  - Communication mechanisms

Software Driven by IML

- GUI for controlling and monitoring instrument
- Commands
  - Validating command arguments
  - Formatting and sending commands to instrument
- Receiving and parsing telemetry from instrument
- Production of Documentation
  - ToolTips
  - On-line reference
  - User manuals

IML Document Type Definition

- Document Type Definition (DTD) specifies some language constraints, e.g.:
  - Commands can have zero or more arguments
  - Arguments must have a name and data type
  - Raw telemetry fields may have a header
- Implications of using DTDs
  - Automatic validation
  - Tools (e.g., XML editors) can use them
  - Some relationships are hard to express
    - Code performs extra validation
- XML Schema (XSDL) will replace DTD
  - strong data typing, subclassing, and constraints expression, far surpassing anything possible with DTDs.

Astronomical Instrument Markup Language (AIML)

- IML is cross domain
  - Astronomy (e.g., telescopes, cameras)
  - Biology (e.g., microscopes)
  - Manufacturing
- IML can be extended if necessary
  - Astronomical Instrument Markup Language
  - Possibilities for instrument-specific extensions
  - Takes advantage of extensibility of XML
  - AIML DTD extends IML DTD

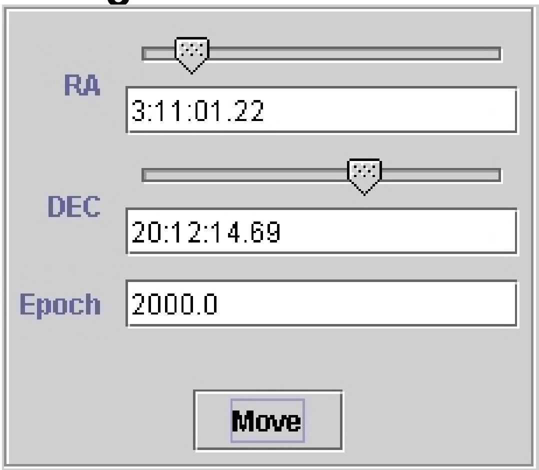
Instrument Description

- Description written in IML using our custom DTD
- InstrumentDescription object
  - Parses IML using Sun’s Java API for XML Parsing
  - Document Object Model (DOM)
  - Instantiates a hierarchy of Java technology-based “Descriptor” objects
  - Hides XML and parser from the rest of the system
    - Could use a different vendor’s parser
    - Could redesign the IML language

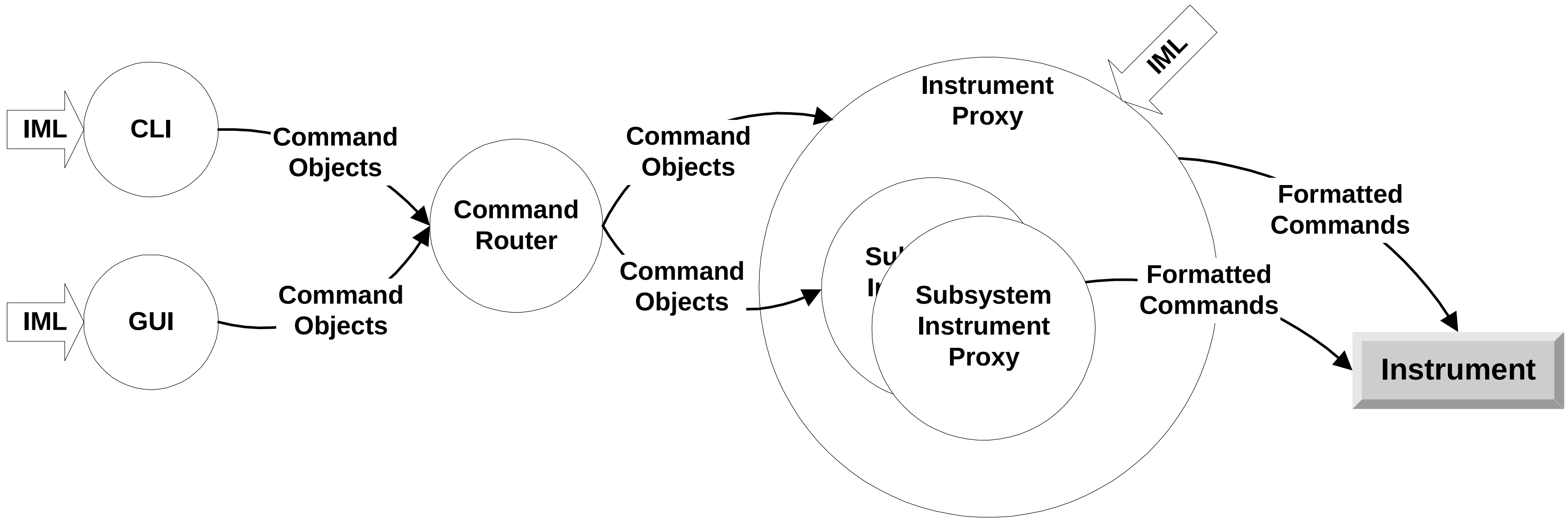
Example IML: Instrument Commands

- IML describes commands and arguments

```
<Command name="Move" >
  <Argument name="RA”
            type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
    <ValidRange low="00:00:00.0" high="23:59:59.99" />
  </Argument>
  <Argument name="DEC”
            type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
    <ValidRange low="-89:59:59.99" high="89:59:59.99" />
  </Argument>
  <Argument name="Epoch”
            type="Float" >
  </Argument>
</Command>
```



Publish and Subscribe



- Commands are published by GUI, Scripts, Command Line Interface (CLI)
- Instrument proxies subscribe for commands specified in IML

Example IML: Formatting Commands

- IML describes how commands are formatted and sent to the instrument

```
<RecordFormat name="Move" terminator="&#10;
       " attributeSeparator=" ">
  <Format name="Command" constant="MOVE" />
  <Format name="RA"      format="%s"   header="RA=" />
  <Format name="DEC"     format="%s"   header="DEC=" />
  <Format name="Epoch"   format="%.1f" header="EPOCH=" />
</RecordFormat>
```

- Results in a String or array of bytes

```
MOVE RA=3:11:01.22 DEC=20:12:14.69 EPOCH=2000.0<CR>
```

Example IML: Formatting Commands

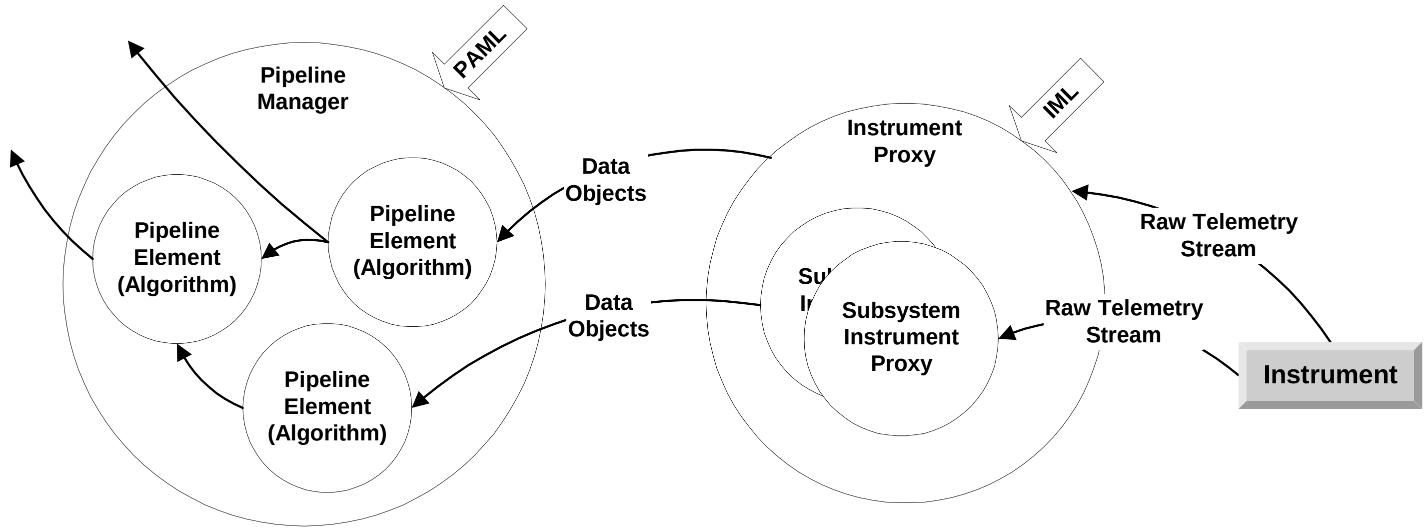
- IML describes how commands are formatted and sent to the instrument

```
<BinaryRecordFormat name="Move"  size="64" >
  <BinaryFormat name="Command" size="4" constant="1100"/>
  <BinaryFormat name="RA"      size="27"/>
  <BinaryFormat name="DEC"     size="27"/>
  <BinaryFormat name=”Epoch"   size="4" />
  <BinaryFormat                size="2" constant="0"/>
</BinaryRecordFormat>
```

- Results in an array of bits

```
11000000001100101100000100101100
00101000011000011101000101001000
```

Instrument Telemetry



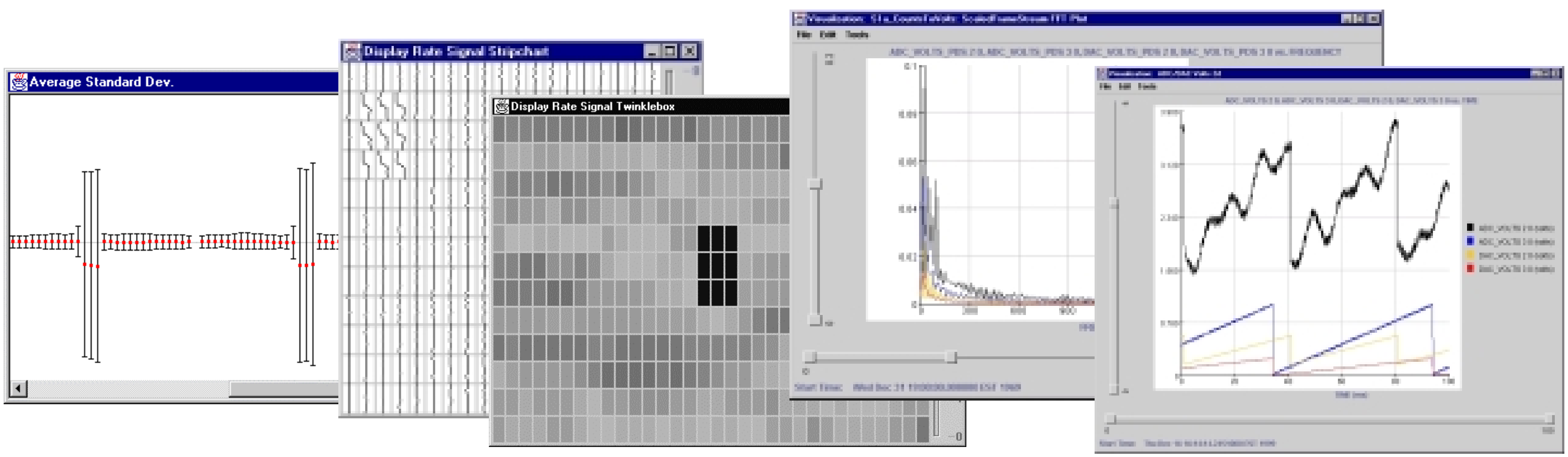
- IML describes how telemetry is formatted
- Instrument proxies receive and parse raw telemetry into objects
- Data Objects are published

Data Analysis Pipeline

- Pipeline is made up of PipelineElements
  - algorithms, archivers, visualization data models
- PipelineDescription (PAML) describes
  - PipelineElements implementation class
  - PipelineElements inputs and outputs
  - PipelineElements commands
- PipelineManager uses description to
  - create algorithms
  - connect subscribers to publishers

Visualizations

- Based on NASA VisAGE architecture
  - Visual Analysis Graphical Environment
  - Visualization toolkit which uses JavaBeans™, Java 2D™, and Java 3D™ technologies
- Visualization data models subscribe to data produced by pipeline elements



Scripting

- Currently using JPython
  - Java implementation of Python
  - Full access to Java packages and Python modules
  - Script objects can extend Java objects and vice versa
  - Designed to allow other scripting languages
- Current capabilities
  - Issue commands
  - Prompt user
  - Add/remove/configure pipeline elements
  - Looping and control flow
  - Could use scripts to extend system

Major Developments: Proof-of-Concept

- Proved that Java technology could remotely control an instrument
- Instrument
  - CARA facility at the South Pole
  - Detector heater controls

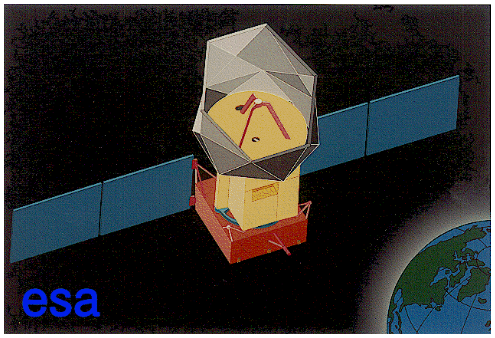
- Proof-of-concept characteristics
  - Application and Web-based applet
  - Dynamic plug-and-play component
  - Distributed architecture
  - JDK™ 1.1 software

Major Developments: XML Prototype

- Proved feasibility of a generic framework with an XML-based instrument description
- Instrument Simulator
  - HAWC: High-resolution Airborne Wideband Camera
  - SOFIA Airborne Observatory
- Prototype characteristics
  - Simple ASCII commands
  - Mix of ASCII and binary telemetry strea
  - Connection to instrument via TCP/IP
  - Windows and Mac
  - JDK™ 1.1.7 software

Major Developments: Engineering Test Instrument

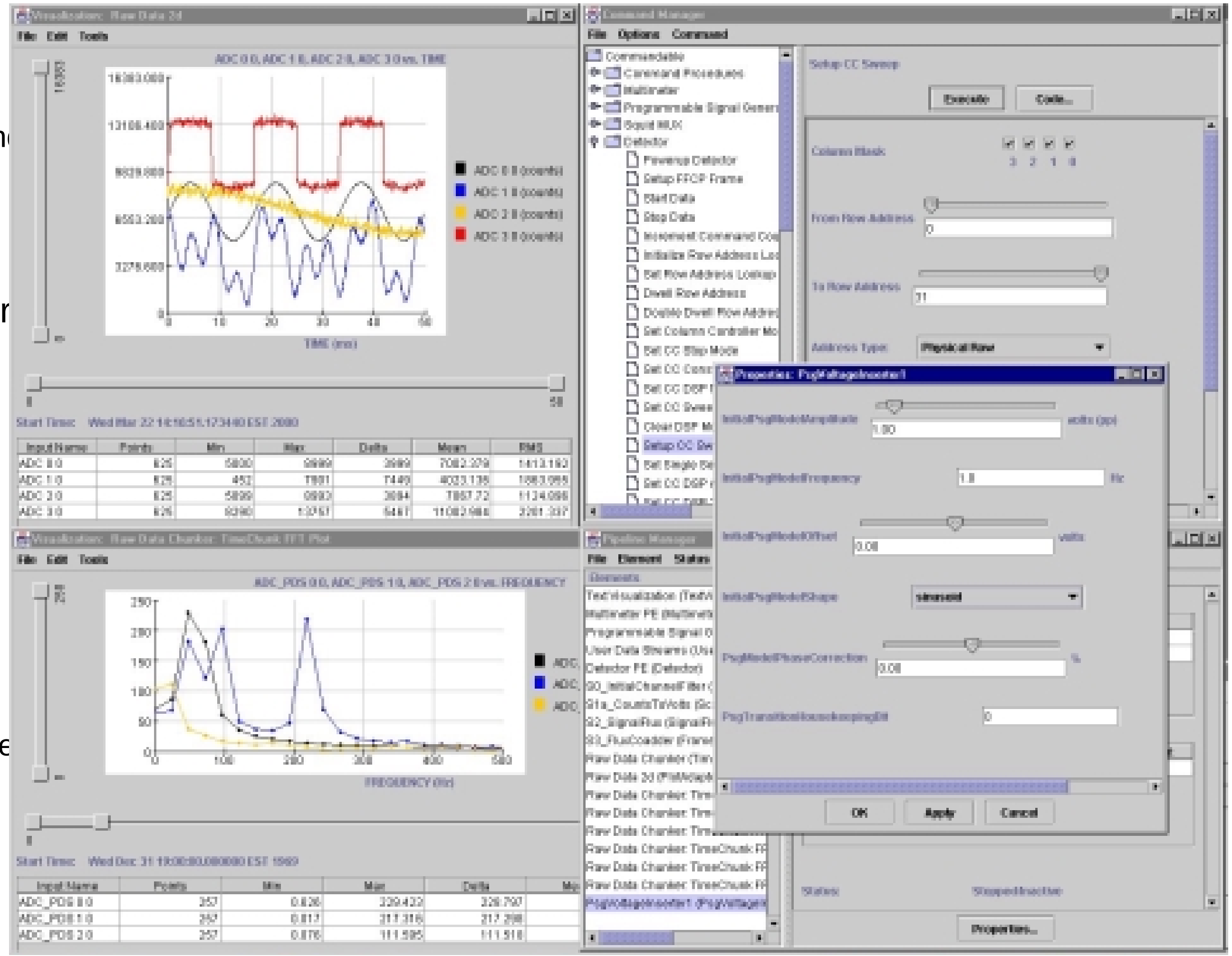
- Completed support for engineering test of instrument
  - Detector downselect candidate for SPIRE
    - Spectral and Photometric Imaging REceiver
  - Space-based telescope for ESA



- Characteristics
  - Binary commands
  - Complex telemetry
  - Connection to instrument via DMA
  - Extremely fast data rate (up to 15 MB / sec)
  - JDK™ 1.2.2 software

IRC Example Screen

- Command Manager
  - sending commands an
scripts
  - specifying arguments
- Pipeline Manager
  - Configuration of Pipeline
  - Properties window for setting algorithm properties (example: Setting the initial state of the Programmable Signal Generator Model Component)
- Visualizations



IRC Project Update

- Proved feasibility
  - Java for remotely controlling instruments
  - Generic framework with XML-based descriptions
- First engineering test of instrument complete
  - Pipeline processes extremely fast data rate (>15 MB/sec)
  - Created Pipeline Algorithm Markup Language Grammar for specifying algorithm characteristics
  - Dynamic, configurable pipeline
  - Scripting capabilities (command procedures)
- Currently enhancing the IRC framework
  - Enhance IML
  - Customizable GUIs
  - External scripting
  - Embeddable

Current Focus

- Instruments
  - HAWC - High-resolution Airborne Wideband Camera (U of C, GSFC)
  - SAFIRE - Submillimeter And Far Infrared Experiment (GSFC)
  - SPIRE - Spectral and Photometric Imaging Receiver (GSFC)
  - SHARC - Submillimeter High Angular Resolution Camera (Caltech)
  - COVIR - Compact Visible & Infrared Imaging Radiometer (GSFC)
- Customers-Partners
  - Infrared Astrophysics Branch (685)
  - Detector Systems Branch (553)
  - Mesoscale Atmospheric Processes Branch (912)
  - Center for Astrophysical Research in Antarctica (CARA)
    - University of Chicago, Carnegie-Mellon University, Caltech, University of New South Wales in Sydney, Australia
  - Stratospheric Observatory for Infrared Astronomy (SOFIA)

The Vision: Instrument Design

- Hardware Engineer defines instrument
  - Uses custom XML editor driven by IML DTD
  - Instrument-specific DTD extensions
- Data Analysis Pipeline
  - Select from library of algorithms
  - New algorithms can be created
    - Description of inputs and outputs enables integration with the pipeline
    - Algorithm implementations: Java™ technology objects, scripts, native code
  - Create and save pipeline configurations

The Vision: Instrument Design

- Graphical User Interface
  - Default GUI may be sufficient for engineering testing
  - Customize GUI
    - Apply existing stylesheet: per instrument and/or per user
    - Define new GUI stylesheet
- Framework will support extensions
  - New data types
  - Unit conversions
  - JavaBean™ GUI components
  - JavaBean Visualizations
  - Performance optimizing delegates

The Vision: Value Proposition

- Generic architecture
- Significant code reuse
- Driven by descriptions
- Anticipate only 10% to 30% customized code
- Savings of 70% to 90% over traditional development paradigms

XML Family of Specifications

- Canonical XML
- Document Object Model (DOM)
- Mathematical Markup Language (MathML)
- Resource Description Framework (RDF)
- Simple API for XML (SAX)
- Synchronized Multimedia Integration Language (SMIL)
- Scalable Vector Graphics (SVG)
- XML Data-Reduced (XDR)
- XForms
- Extensible HyperText Markup Language (XHTML)
- XML Inclusions (XInclude)
- XML Link Language (XLink)
- XML Information Sets (XML Infosets)
- XML Namespaces
- XML Schema (XSDL)
- XML Path Language (XPath)
- XML Query Language
- Extensible Stylesheet Language (XSL)
- Extensible Stylesheet Language Formatting Objects (XSL-FO)
- Extensible Stylesheet Language Transformations (XSLT)

XML Specifications

- Canonical XML
  - subset of a XML document that defines its logical structure; used to determine logical equivalence of 2 documents.
- Cascading Style Sheets (CSS)
  - pre-date XML but can be used with XML for simple styling at the element level.
- Document Object Model (DOM)
  - platform- and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure and style of document (as represented by a tree view)
- Mathematical Markup Language (MathML)
  - textual description of math formulae and symbols in XML syntax; supported today with IBM techexplorer browser plugin
- Resource Description Framework (RDF)
  - foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web.

XML Specifications

- Simple API for XML (SAX)
  - non-W3C, grass roots effort to define an event-based application programming interface to XML that is language and parser independent.
- Synchronized Multimedia Integration Language (SMIL, pronounced "smile")
  - describes audio and/or video presentations in simple textual markup using a single timeline.
- Scalable Vector Graphics (SVG)
  - permits resolution independent textual descriptions of graphical objects, as well as animation, scripting events, spatial translations, filter effects, etc.; tightly integrated with other XML specifications, such as DOM, CSS, XSLT, XLink, and XHTML
- XML Data-Reduced (XDR)
  - was a variation on the Microsoft proposal to add data types to XML; early ancestor of what is now XML Schema; still used by MIcrosoft's BizTalk.org
- XForms
  - "the next generation of Web forms"; goal is clear separation of the user interface from the data and logic; can function as XHTML modules or can be integrated with other XML languages

XML Specifications

- Extensible HyperText Markup Language (XHTML)
  - reformulation of HTML 4.01 in XML syntax, plus the dividing of HTML into modules that can be implemented separately on different devices; many XHTML specs including XHTML 1.0, XHTML 1.1, XHTML 2.0, XHTML Basic, XHTML Modularization, XHTML Events, etc.
- XML Inclusions (XInclude)
  - "specifies a processing model and syntax for general purpose inclusion. Inclusion is accomplished by merging a number of XML Infosets into a single composite Infoset."
- XML Link Language (XLink)
  - defines how to create complex interconnections between documents and between elements, as well as the simple unidirectional linking provided by HTML.
- Extensible Markup Language (XML)
  - syntax for describing data in a hierarchy that can be easily parsed and validate; platform-independent data description language.
- XML Information Sets (XML Infosets)
  - describes an abstract data set containing the information available from a well-formed XML document that follows the Namespaces spec.

XML Specifications

- XML Namespaces
  - define a convention based on URI (uniform resource identifiers) for uniquely qualifying element and attribute names to avoid names collisions when multiple XML grammars are used in one document
- XML Schema
  - also known as XSDL (XML Schema Definition Language), is a way to describe the content and data model of a particular XML vocabulary with strong data typing, subclassing, and constraints expression, far surpassing anything possible with DTDs.
- XML Path Language (XPath)
  - syntax for defining a path within the DOM hierarchy of a document, for addressing specific elements and/or attributes; used by both XSLT and XPointer.
- XML Query Language
  - generic name applied to several different approaches to SQL-like or XPath-like searches within an XML document; active XML Query Working Group.

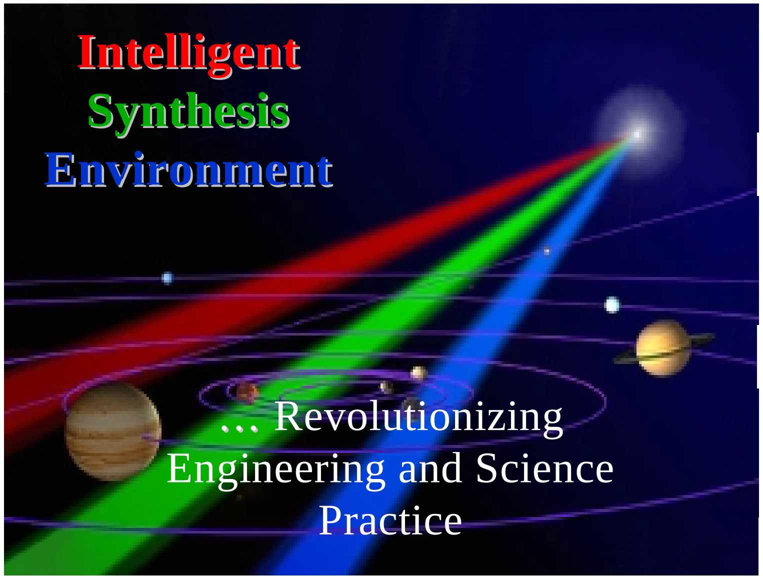
XML Specifications

- Extensible Stylesheet Language (XSL)
  - is actually a two-part specification, including XSL-FO and XSLT; confusingly, XSL sometimes means either of the two parts or both, but most likely refers to the Formatting Objects portion.
- Extensible Stylesheet Language Formatting Objects (XSL-FO)
  - the major portion of the XSL specification that describes sophisticated layout and presentation mechanisms much like those used in the desktop publishing world.
- Extensible Stylesheet Language Transformations (XSLT)
  - that part (Section 2) of the XSL specification that defines how to transform XML documents into other XML documents; defines a mechanism for filtering, sorting, or otherwise transforming an XML source tree into a result tree.

For More Information

- NASA/Goddard IRC website (prototype, papers, presentations, DTD):
  - http://pioneer.gsfc.nasa.gov/public/irc
- NASA/Goddard XML for Astronomy website
  - http://pioneer.gsfc.nasa.gov/public/xml
- Listed on XML.org
  - http://www.xml.org/xmlorg_catalog.htm
- Scientific American, May 1999 - XML and the Second-Generation Web article
  - http://www.sciam.com/1999/0599issue/0599bosakbox4.html
- Two XML books written by Simon St. Laurent
  - Inside XML DTDs: Scientific and Technical, July 1999
  - XML: A Primer, Second Edition, September 1999

Integrated Exploration and Science

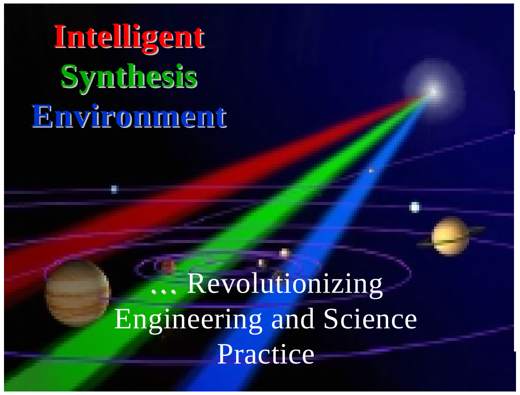


Mars Exploration Application

Dr. Eric M. De Jong

Integrated Exploration and Science, Science Lead
Solar System Visualization Project, Project Scientist
Eric.M.DeJong@jpl.nasa.gov (818) 354-0302

Integrated Exploration and Science



Mars Exploration Application

Dr. Eric M. De Jong

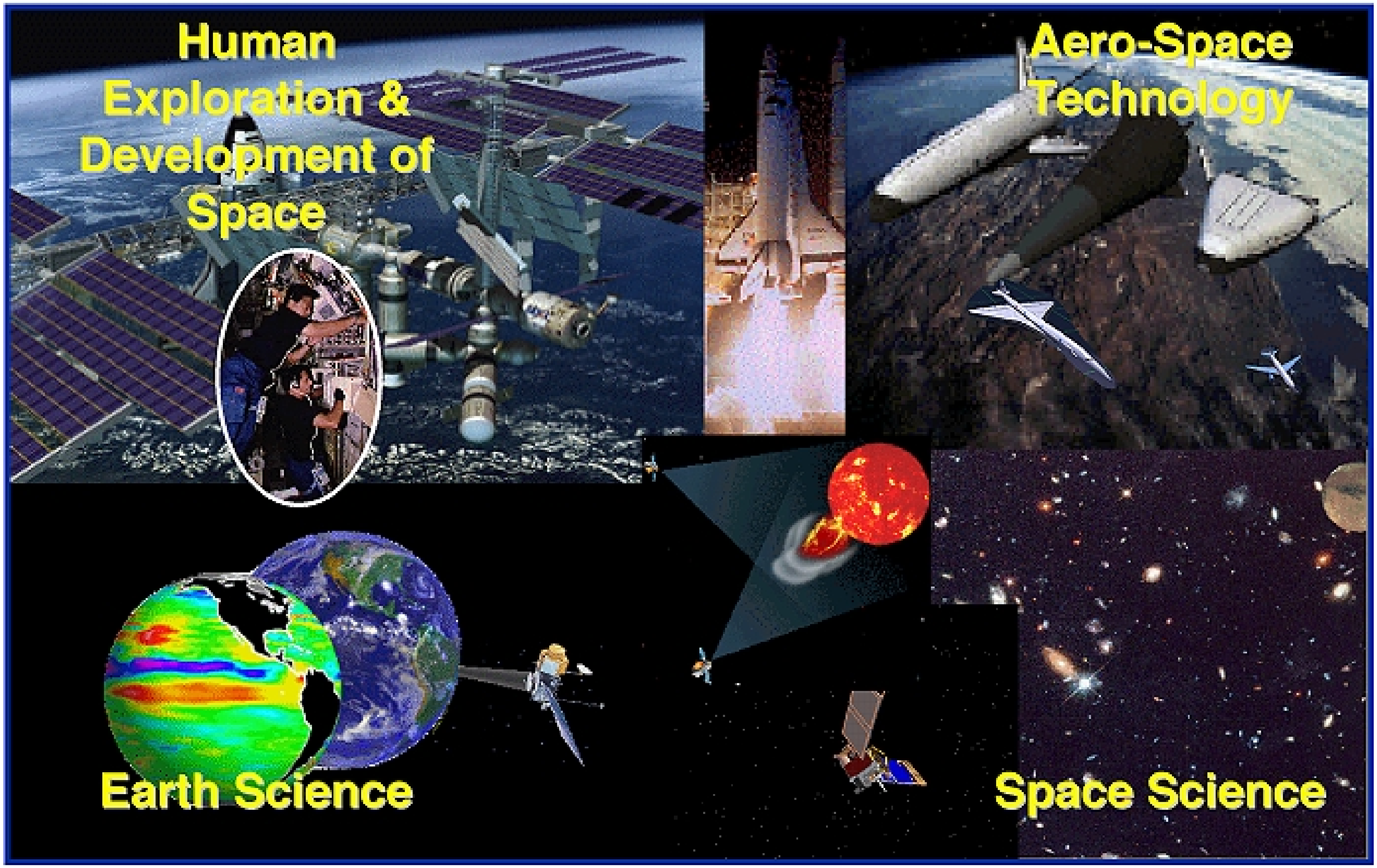
Integrated Exploration and Science, Science Lead
Solar System Visualization Project, Project Scientist
Eric.M.DeJong@jpl.nasa.gov (818) 354-0302

Director ISE Program: Dr. John B. Malone, LaRC

- JPL Lead Positions:
  - Patricia Liggett — ISE JPL Deputy Program Manager
  - John Peterson — Special Assistant for Science Integration
  - Eric De Jong — Integrated Exploration and Science Application, Science Lead
  - John Baker — Integrated Exploration and Science Application, Engineering Lead
  - Ed Chow — CEE Collaborative Infrastructure Development Sub-Element Team Lead
  - Meemong Lee — RSST Simulation and Modeling Tools Sub-Element Team Lead

- JPL Supporting Positions:
  - Hamid Habib-Agahi, Bob Shishko, (CRMT)
  - Keith Warfield (CRMT)
  - Dave Smith (RCCTE)
  - Steve Wall (LCIV)
  - Richard Weidner (RSST)

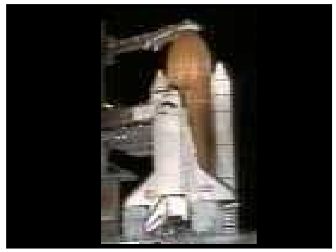
NASA's Four Strategic Enterprises



NASA's Future Mission Challenges

NASA's enterprises envision a myriad of highly complex, first-of-a-kind, missions *which must be developed and executed within reduced budget, workforce and time constraints*

Single Stage to Orbit Launch Vehicles

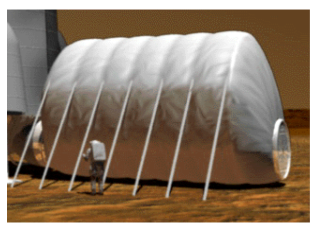


Shuttle Upgrades/ISS Operations

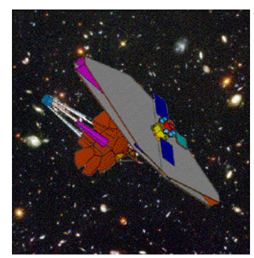
Earth Science Sensing Fleet

Planetary Sample Return

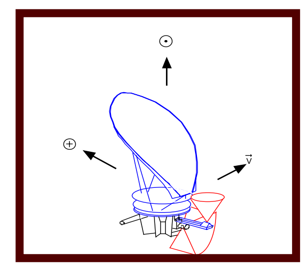
Advanced Aircraft Concepts



Human Exploration

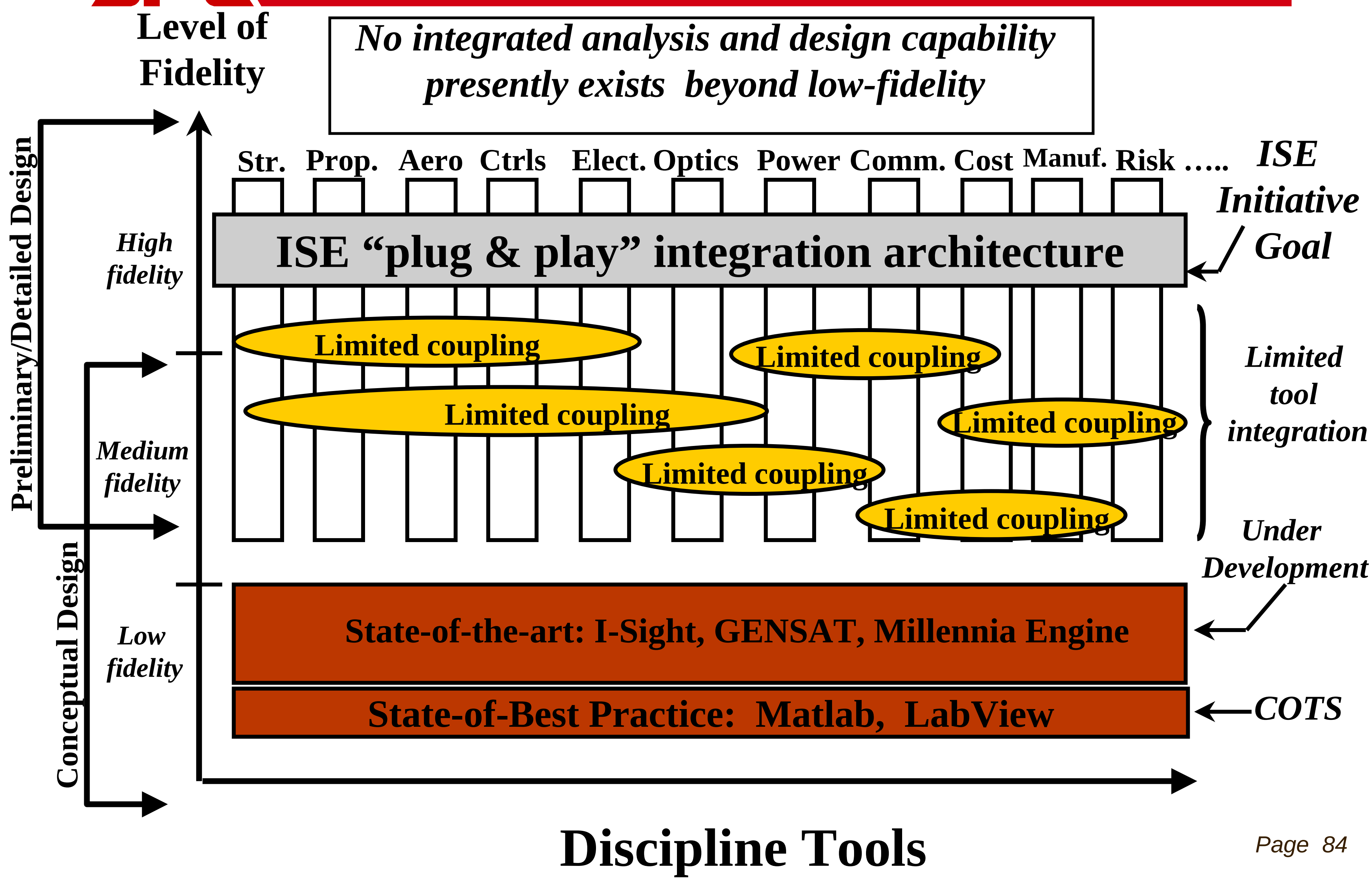


Next Generation Astronomy



Near-Sun Measurements

Future Engineers Need A Highly Integrated Analysis and Design Capability



Critical Engineering and Science Challenges As Identified by Industry Workshop*

- **Category: Design Tools**
  - Lack of accurate costing and risk prediction methods
  - Long model development and simulation time
- **Category: Design process**
  - Design development cycle remains essentially sequential 10 years after concurrent engineering philosophy adopted by government and industry
  - Design and development process steps and their interactions not well understood leading to large design iterations
  - Lack of tool integration except at a conceptual design level
  - Current design process relies too heavily on testing
- **Category: Insertion of new technology**
  - Takes many years for new technology to enter practice
- **Category: Design creativity**
  - Design creativity inhibited by clumsy processes, empirical tools and lack of collaboration
  - Little collaboration between scientists, engineers, operations and training personnel

* Code R sponsored Design Tools Roadmapping Workshop, August 17-19, 1998

Engineering Environments Must Change to Meet Challenges of 21st Century Mission

Digital Life & Full Virtual Product

Where we Need to be

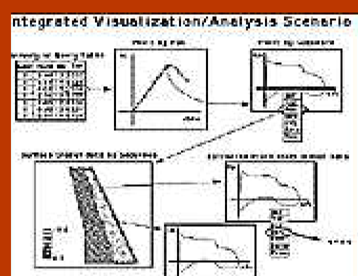
ISE

Digital Process

Digital Mock-up

Where we are now

Pre-CAD



2D Drawings

3D CAD

Where we've been

ISE Vision and Long-Term Goal

Vision

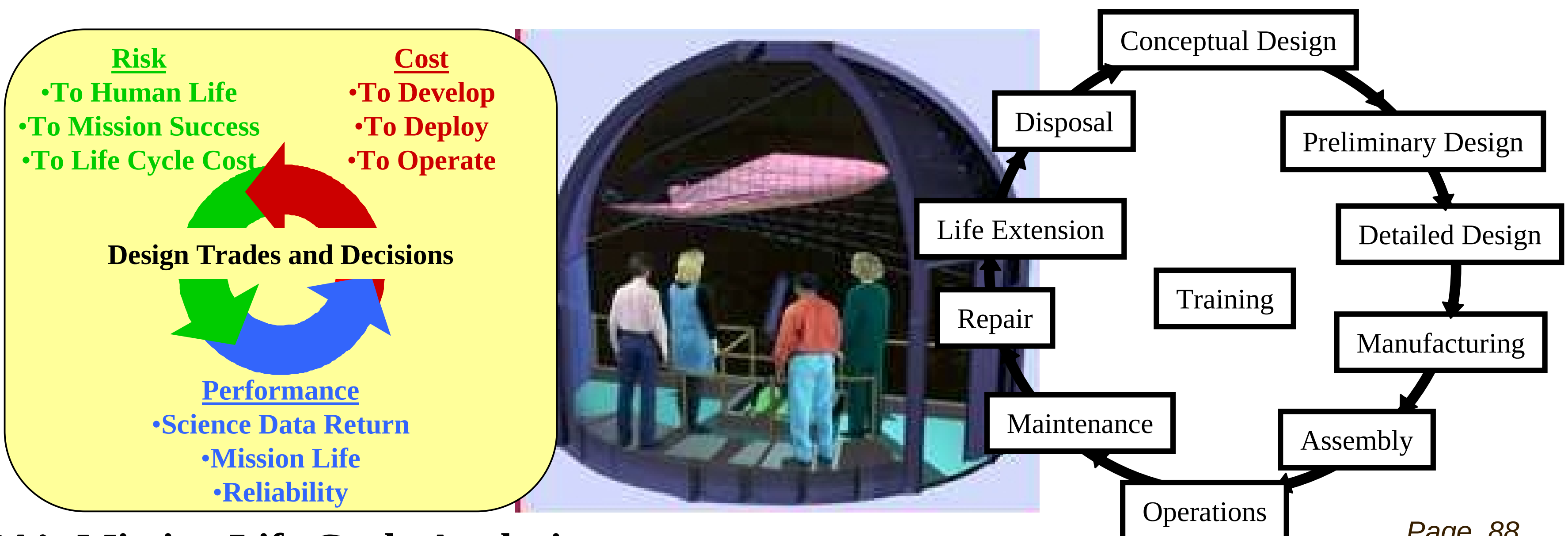
To effect a cultural change that integrates into practice widely-distributed science, technology and engineering teams to rapidly create innovative, affordable products

Long-Term Goal

To develop the capability for personnel at dispersed geographic locations to work together in a virtual environment, using computer simulations to model the complete life-cycle of a product/mission with near real-time response time before commitments are made to produce physical products

A “Holodeck-Like” Design Capability for 21st Century Science and Engineering Teams

- Close coupling of Science Mission Requirements and Engineering product/platform solutions prior to acquisition “go-ahead”
- Near real-time design changes with resulting impacts to all elements of the product/mission life-cycle
- Engineers/Scientists can experience complete immersion and sensory feedback within the design environment
- A true multidisciplinary design environment with instantaneous sharing of data and knowledge between Science and Engineering experts



NASA’s Mission Life Cycle Analysis

ISE Attributes

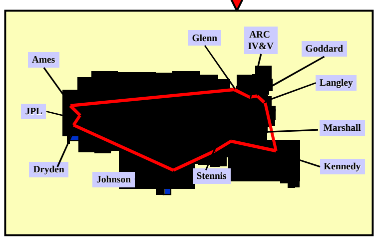
- A national network of geographically distributed engineering and science capabilities
- Progressively infused with advanced methods and tools from R&D Laboratories
- Specific hardware, methods and tools driven by Enterprise applications and training needs
- Full life-cycle design and analysis for new missions and systems
- Combines intelligent computing methods, such as neural networks, genetic algorithms and fuzzy logic with physics-based simulations to provide rapid and accurate data/knowledge creation
- Heterogeneous hardware and software mix
- Intelligent and adaptive re-configurable capability
- Automated intelligent selection of hardware, tools and methods
- Intelligent decision making aids
- Cooperative linking of facilities
- Cooperative agreements and contracts with engineering software vendors, aerospace industry, non-aerospace industry, and universities

Achieving a Long-Term Vision in a Changing World

Vision: "… a cultural change that integrates into practice widely-distributed science, technology and engineering teams to rapidly create innovative, affordable products", Dan Goldin, 1998

1998 2000 2002 2004 2006 2008 2010 2012 2014 2016

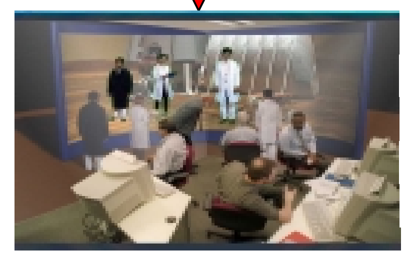
Start — Ø-1 Completion — Ø-2 Completion — Realization



ISE Planning Team

Enabled

Applied



Ubiquitous

Implementation

Operation

<u>At conception</u>

Access: T-1/T-3 Communication Links
Users: < 1% of Hardware Design Engineers (est.)
Participation: Technology Efforts/Single Companies
Systems: Work Stations, Interactive Screens, Video-Conference
Construct: Wrapped Software, Legacy Models, Stand-Alone Databases

<u>At completion</u>

Wide-Band Optical/Satellite
100% of All Design/Development Engineers
Reconfigurable International Distributed Teams
Information Power Grid/Affordable Immersion
Plug-in Software, Intelligent Modules, Online Knowledge Bases

The ISE 5-year challenge: A foundation and framework to enable the 15-year vision

ISE Far-Term Program Roadmap

Goddard Space Flight Center
JPL

Vision: To effect a cultural change that integrates widely-distributed science, technology and engineering teams to rapidly create innovative, affordable products.

ISE Initiative

Program Elements	Near-Term 1999-2001	Mid-Term 2002-2004	Far-Term 2005-2015
• **Rapid Synthesis and Simulation Tools** • **Cost and Risk Management Technology** • **Life-Cycle Integration and Validation** • **Collaborative Engineering Environment** • **Revolutionize Cultural Change, Training and Education**	• Integrated, "Best-in-Practice" engineering tools • Full mission life cycle cost and risk analysis • Geographically distributed, electronically collaborating teams • Upgrade skills of technical employees • Collaborative Video Conferencing	• Networked high fidelity design, non-traditional tools • Design intelligence for mission cost and risk optimization • State of the art practice in all NASA engineering • Innovative university education programs	• Engineers and scientists collaborating on virtual and real missions in a networked immersive environment • Confident capability for accurate cost and risk trades on complex first-of-a-kind missions • Collaborative immersion and virtual co-location

ISE 5-Year Goal

To develop the capability for personnel at dispersed geographic locations to work together in a virtual environment, using computer simulations to rapidly model the complete life-cycle of a product/mission before commitments are made to produce physical products

Agreement on Technical Content and Interactions of Administrator's "Pathway" Initiatives

"Pathway to the Future"

(Administrator's Vision)

LaRC Lead

Intelligent Synthesis Environment (ISE)

- Distributed Collaborative Engineering Environment
- Immersive Multimedia VR Environment
- New Analysis Tools
  - Engineering
  - Life Cycle Cost & Risk
- Large-scale Applications

Functional Initiatives integrated through NASA Chief Technologist

ISE Requirements

IS Products

ARC Lead

Intelligent Systems

- Automated Reasoning: "Thinking Systems"
- Human-Centered Computing
- Intelligent Use of Data: "Knowledge Generation"
- Revolutionary Computing

Revolution in NASA's engineering & science capability and infrastructure

Elements of the ISE Functional Initiative

Rapid Synthesis and Simulation Tools

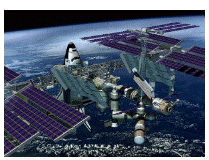
Developing advanced intelligence-based engineering and science simulation tools for analysis and design from concept through disposal and synthesis tools for seamless coupling of diverse discipline tools

Cost and Risk Management Technology

Develop advanced cost analysis and risk tools in a unified framework covering end-to-end mission design, and compatible with design and analysis tools for fully integrated life cycle simulations.

Life-Cycle Integration and Validation

Developing integration methods, smart interfaces and frameworks to achieve seamless "plug and play" integrated design and analysis, and assessment, validation and demonstration of ISE technologies.



Collaborative Engineering Environment

Advancing the state of practice and inserting the state of the art collaborative infrastructure and applied design and analysis capabilities into enterprise use.

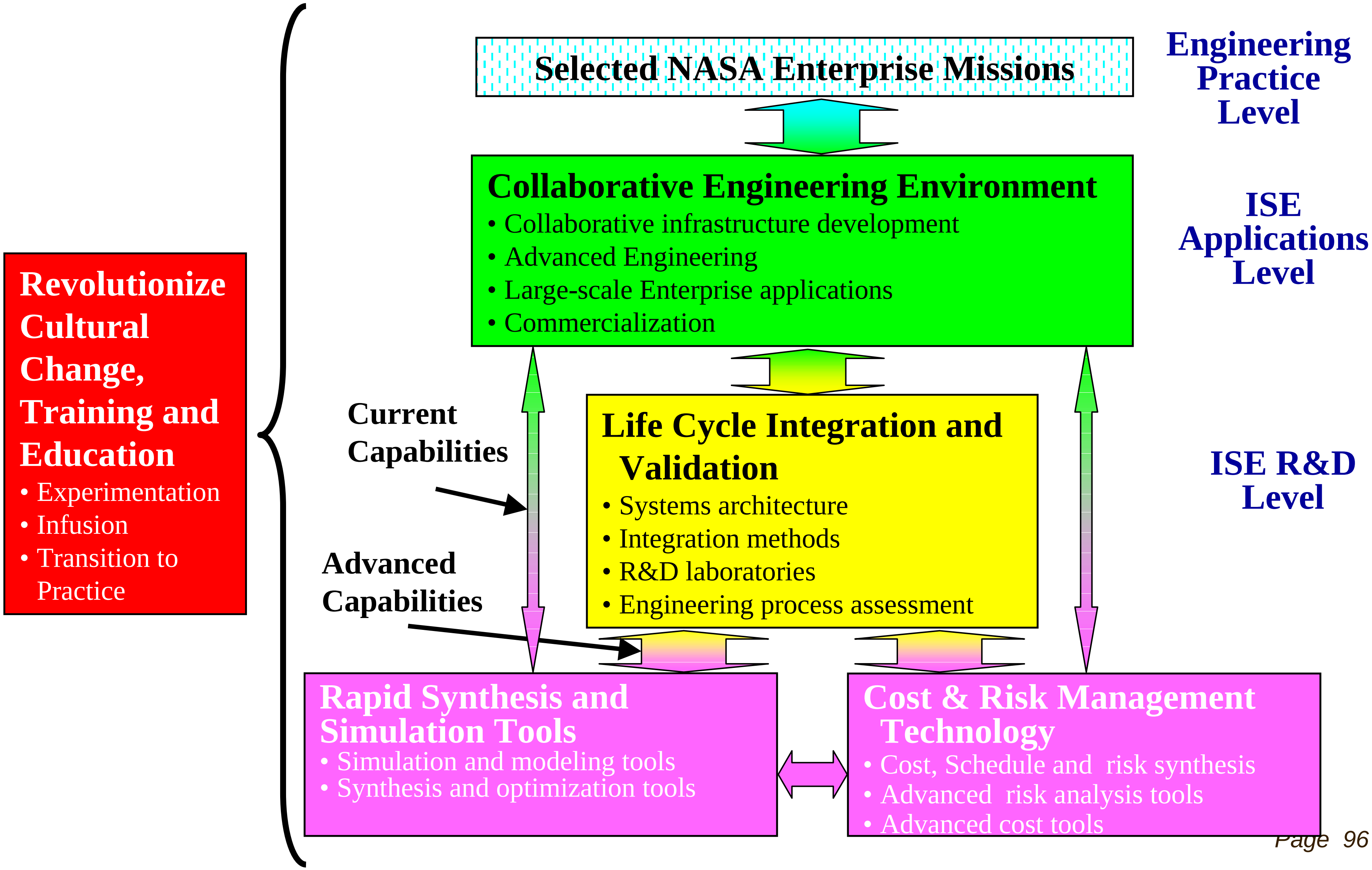
Revolutionize Cultural Change, Training and Education

Changing the engineering culture to take full advantage of advanced tools and environments and developing distributed active learning and training collaborative environment

Level 1 ISE Initiative Requirements

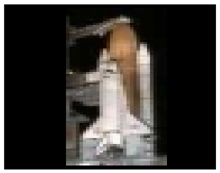
- Design and analysis tools
  - Demonstrate factor of 10 reduction in time over baseline for multidisciplinary system and process design and analysis functions for all Large-Scale Applications without loss of baseline fidelity, or alternatively, a factor of 10 increase in fidelity with no loss of speed
- Cost and risk management capability
  - Demonstrate capability to compute system and process costs and risks with 5 times baseline fidelity in one-fifth baseline time for all Large-Scale Applications
- Mission life-cycle integration
  - Demonstrate capability, applicable to all enterprises, to trade performance, cost, and risk for complete mission life-cycles with 10 times baseline fidelity or alternatively, a factor of 10 increase in speed with no loss of fidelity
- Engineering and science practice in NASA Enterprises
  - For each enterprise, demonstrate design cycle time of one-fourth of enterprise baseline for complete life-cycle of a new mission concept in a geographically distributed collaborative team of NASA, industry, and academia
- Engineering and science culture and creative processes
  - Demonstrate factor of 10 increase over baseline in percentage of participants in NASA enterprise science and engineering design, development, and management activities using geographically distributed electronic collaboration

Relationship Between ISE Elements



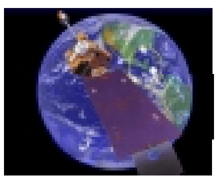
Large-Scale Applications (LSAs)

Reusable Space Transportation Systems



Shuttle/International Space Station

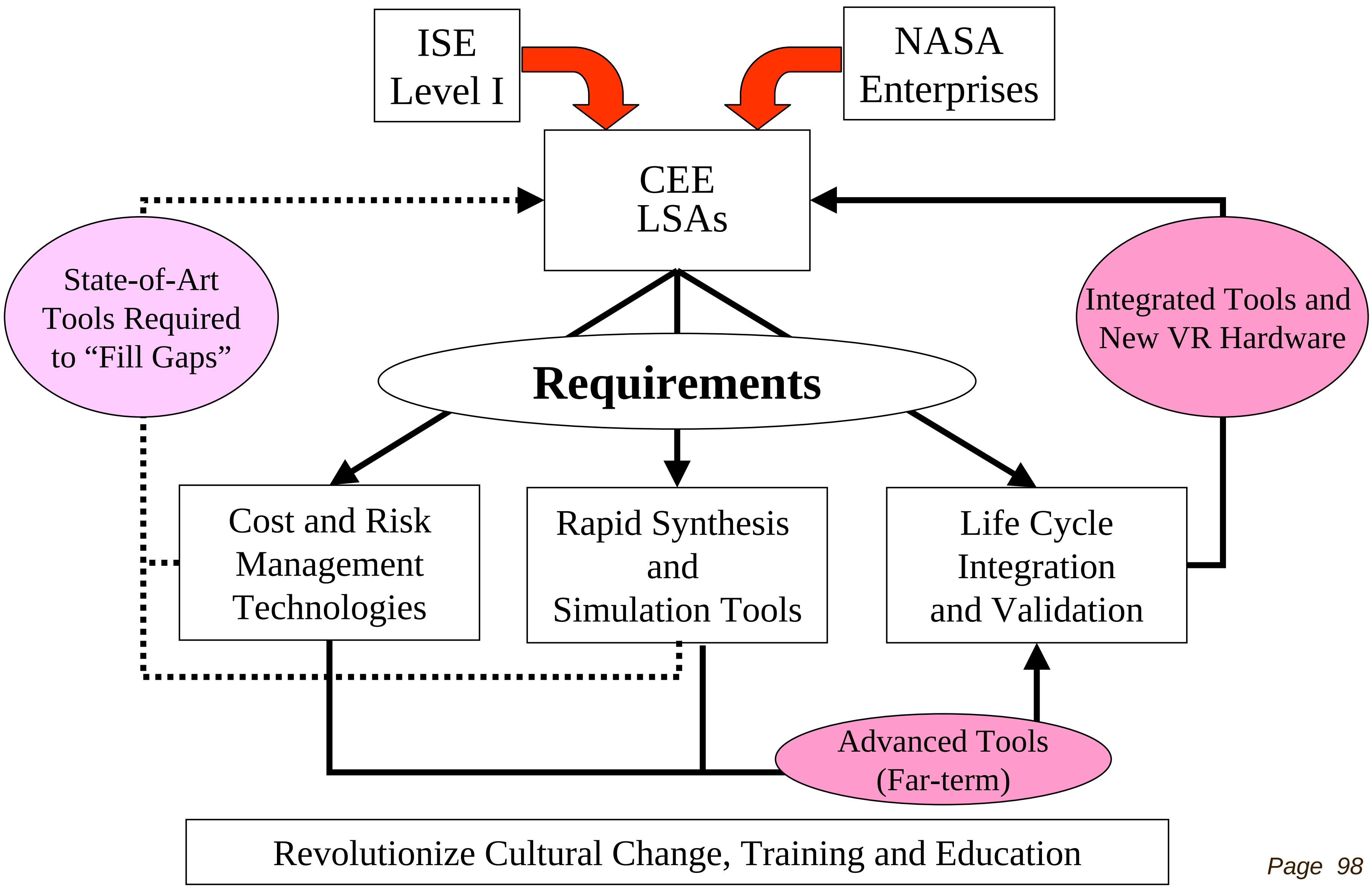
Integrated Exploration and Science



Advanced Earth Observation

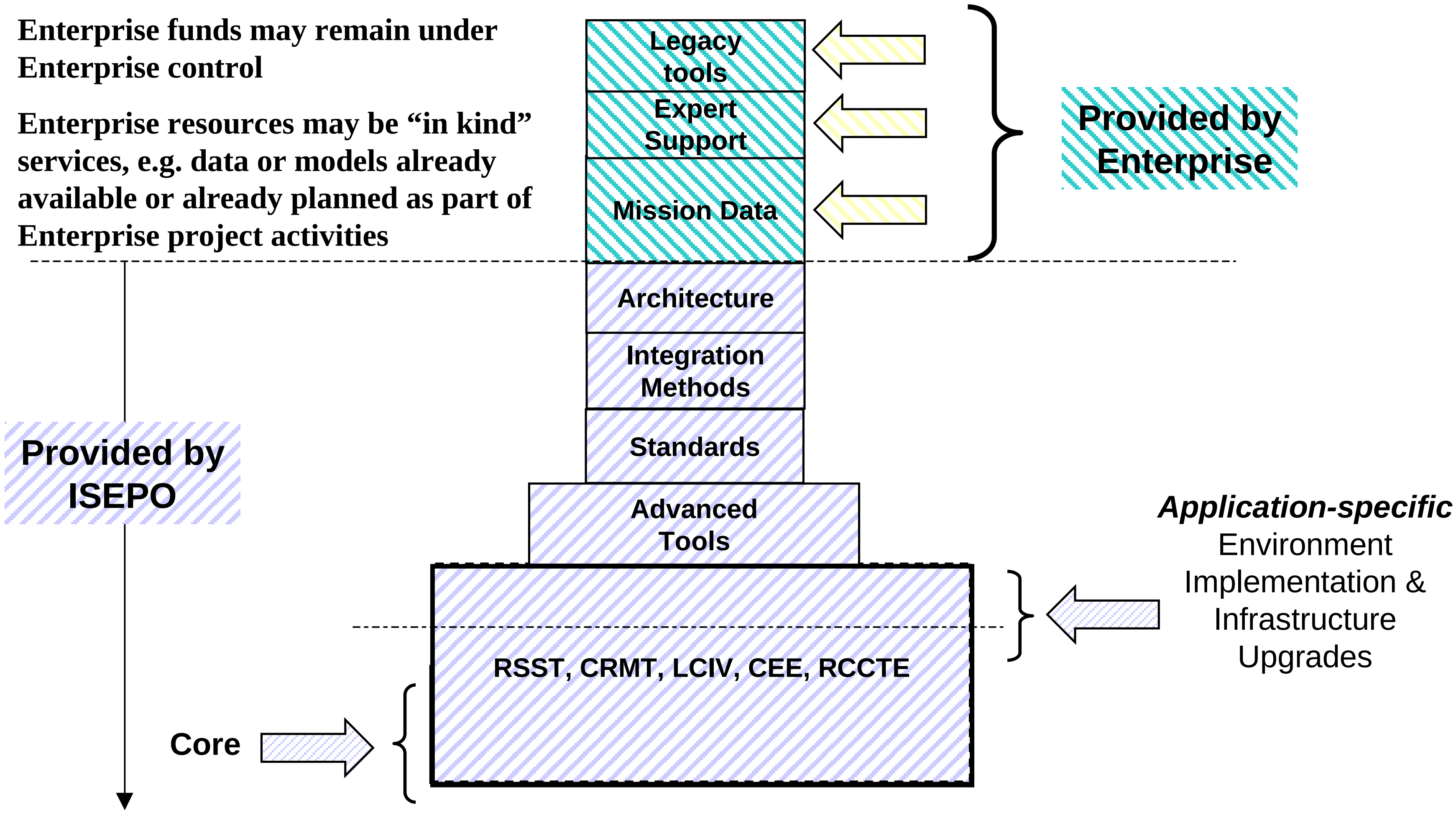
- ***Tightly couples NASA R&D Center research products to Development Centers Needs***
- ***An integration of computer hardware, software and facilities that enables the development of a design/analysis capability focused on specific mission needs***

ISE Planning Requirements Flow



ISEPO and Enterprise Jointly Support LSA's

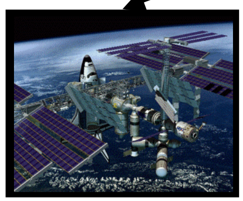
- Enterprise funds may remain under Enterprise control
- Enterprise resources may be "in kind" services, e.g. data or models already available or already planned as part of Enterprise project activities

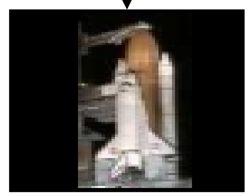


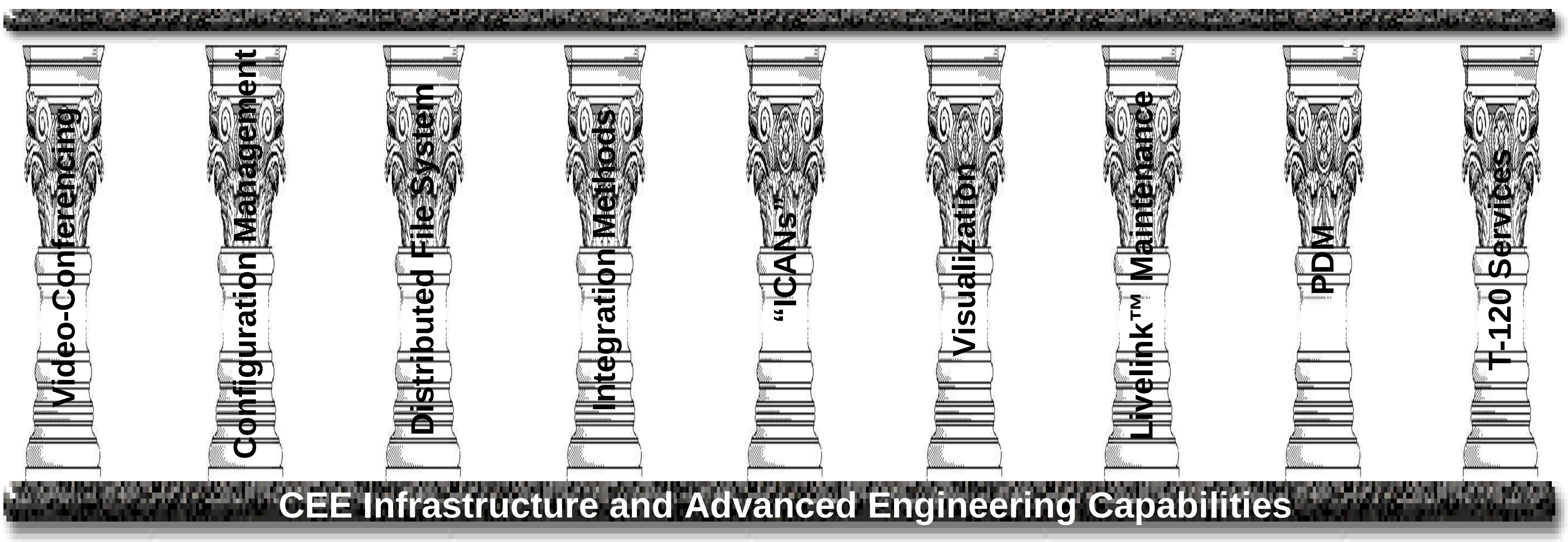
Actual Enterprise contributions to LSAs determined by application planning teams and negotiated with Enterprise management

Common Services Provided to Applications

CEE Investment in Infrastructure and Advanced Engineering

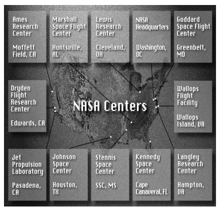


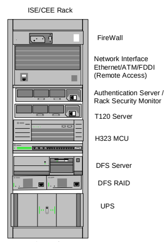


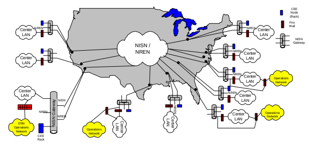


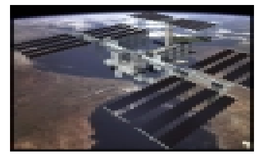
Collaborative Infrastructure Development Roadmap

Now | 1999 | 2000 | 2001 | 2002 | 2003 | 2004



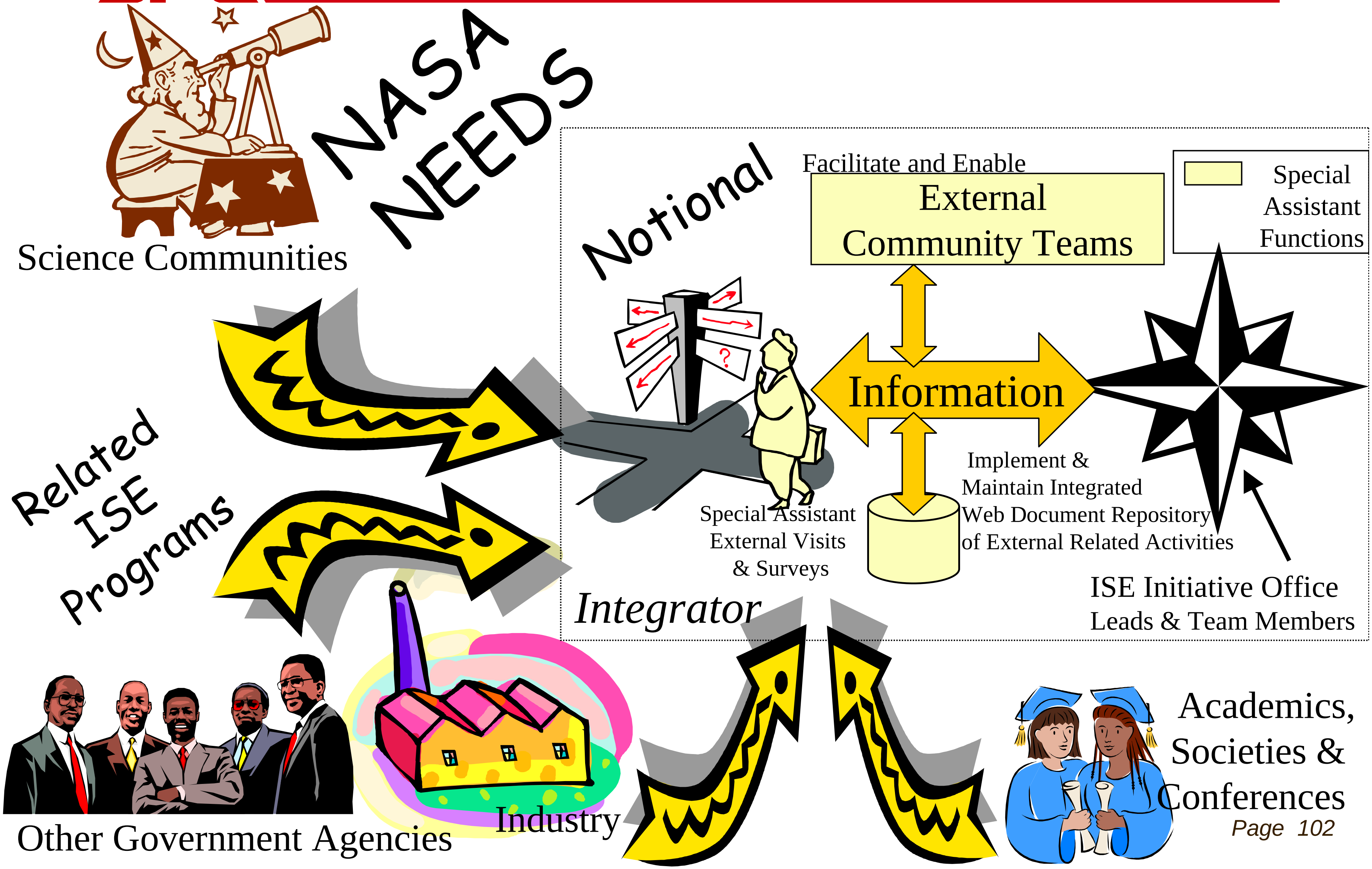




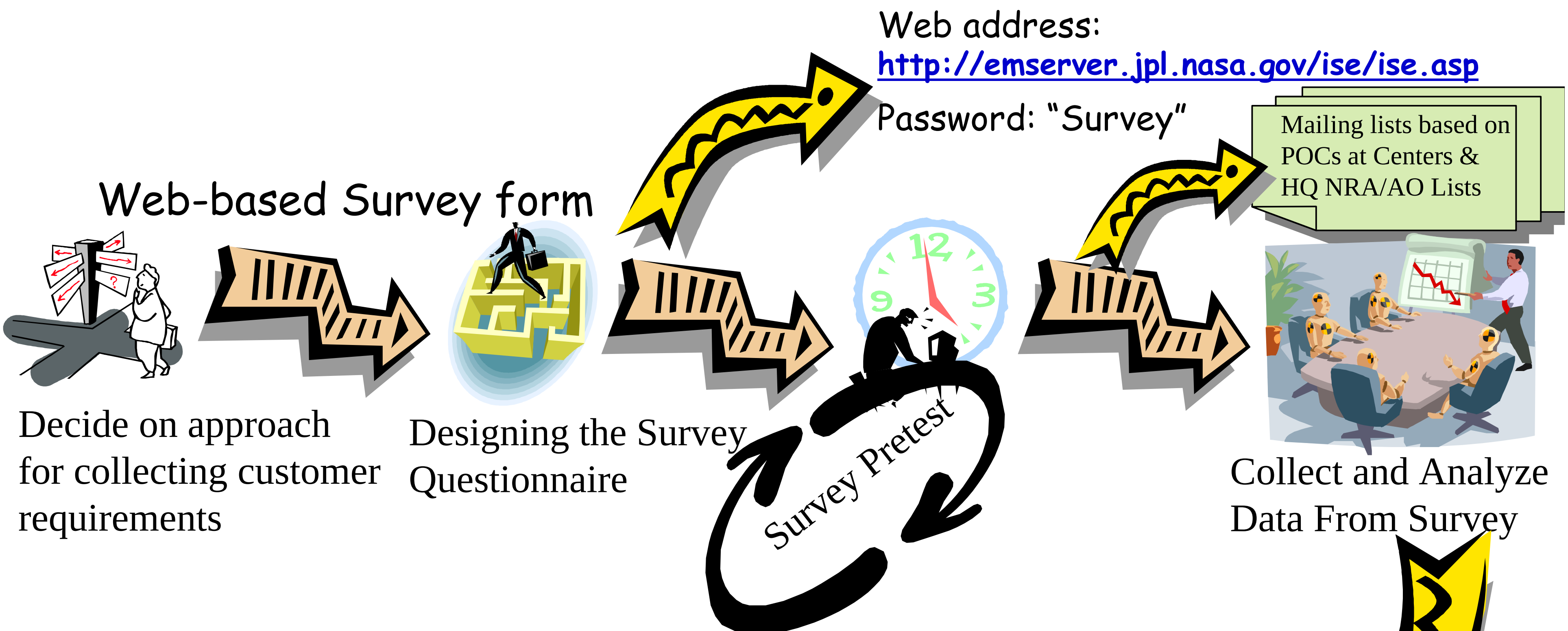


CEE	CEE rooms	CEE ICAN Development & Deployment	CEE Information Infrastructure	CEE Information Infrastructure for Operational Support	ISE Technology Empowered CEE Information Infrastructure	Integrated ISE mission planning, development, operation environment
CEE Services	Video conferencing, NetMeeting,	Secured Networking, Video Conf.,, NetMeeting, LiveLink, Multi-time zone calendaring, Limited DFS, EMS, VOD, Limited H.323	Full DFS, Redundant Services, PDM pilot deployment, H.323 desktop pilot deployment, Application framework for ISE technology insertion. & basic immersive product visualization	Operation ready network security, Multi-center mission oriented PDM system, Knowledge navigation system, Immersive multimedia node, ISE distributed modeling support	CEE infrastructure with customer ready ISE simulation development tools, High performance CEE networking and file systems that can support ISE advanced tools	Immersive product and mission visualization capability, User transparent virtual planning, simulation, and operations.
Success Criteria	At least one compatible CEE room at each center	Deploy initial ICAN at NASA centers, Gather customer requirements, ISE user use CEE for development	"Spots" mission support successes	Use CEE to support enterprise applications development process including operations	Use ISE technology on CEE to demonstrate a mission with a development cycle <18 months	Integrated End-to-End mission development/ support environment.

Integration Concept



Requirements Collection Approach



Survey data will provide team with preliminary directions for target ISE Initiative technologies and infrastructure and in planning the design and content for a workshop in 2000

Integrated Exploration and Science

Dr. Eric De Jong, John Baker
Jet Propulsion Laboratory
(818) 354-0302/9464
Eric.M.DeJong@jpl.nasa.gov
John.D.Baker@jpl.nasa.gov

Deb Neubek
Johnson Space Center
(281) 483-9416
deborah.j.neubek1@jsc.nasa.gov

Integrated Exploration and Science (IE&S) Application Summary

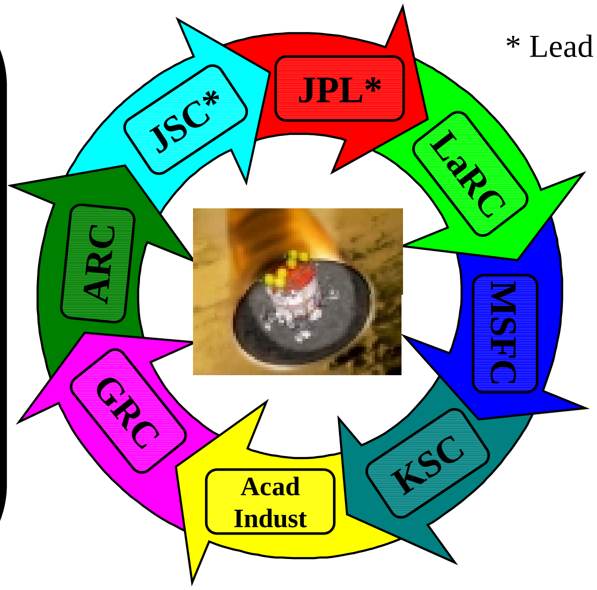
Mars Exploration - **Develop and adapt a collaborative design environment for Mars missions to enable an end-to-end mission analysis and concept design in support of the Next Decade Planning/Robotic Outposts, Space Science Enterprise and Human Exploration and Development of Space activities. The following elements & organizations are included**:

- **Mission-System Architecture**
  - Orbiter, lander, rover, surface, ascent, earth return & entry vehicles *(JPL, JSC,GRC, KSC, MSFC, LaRC, HRET, LPI, industry, academia)*
  - HEDS payloads (in-situ, dust, etc.) *(JSC, HRET, LPI, industry, academia)*
  - Science Payloads *(JPL, LPI, academia)*
- **Design of Life-Cycle Process**
  - Development, Launch & Cruise *(MSFC, JSC, Lockheed Martin, SAIC)*
  - Aero-descent *(ARC, JSC, LaRC)*
  - Surface Exploration (Lander, Payloads) *(JPL, JSC, ARC, GRC)*
  - Ascent Vehicle and Return Orbiter *(JPL, CNES)*
  - Earth Entry Vehicle *(LaRC, JSC)*
- **Mars Environment**
  - Atmospheric, weather/climate, and terrain models *(ARC,JPL)*

Integrated Exploration and Science

SCOPE

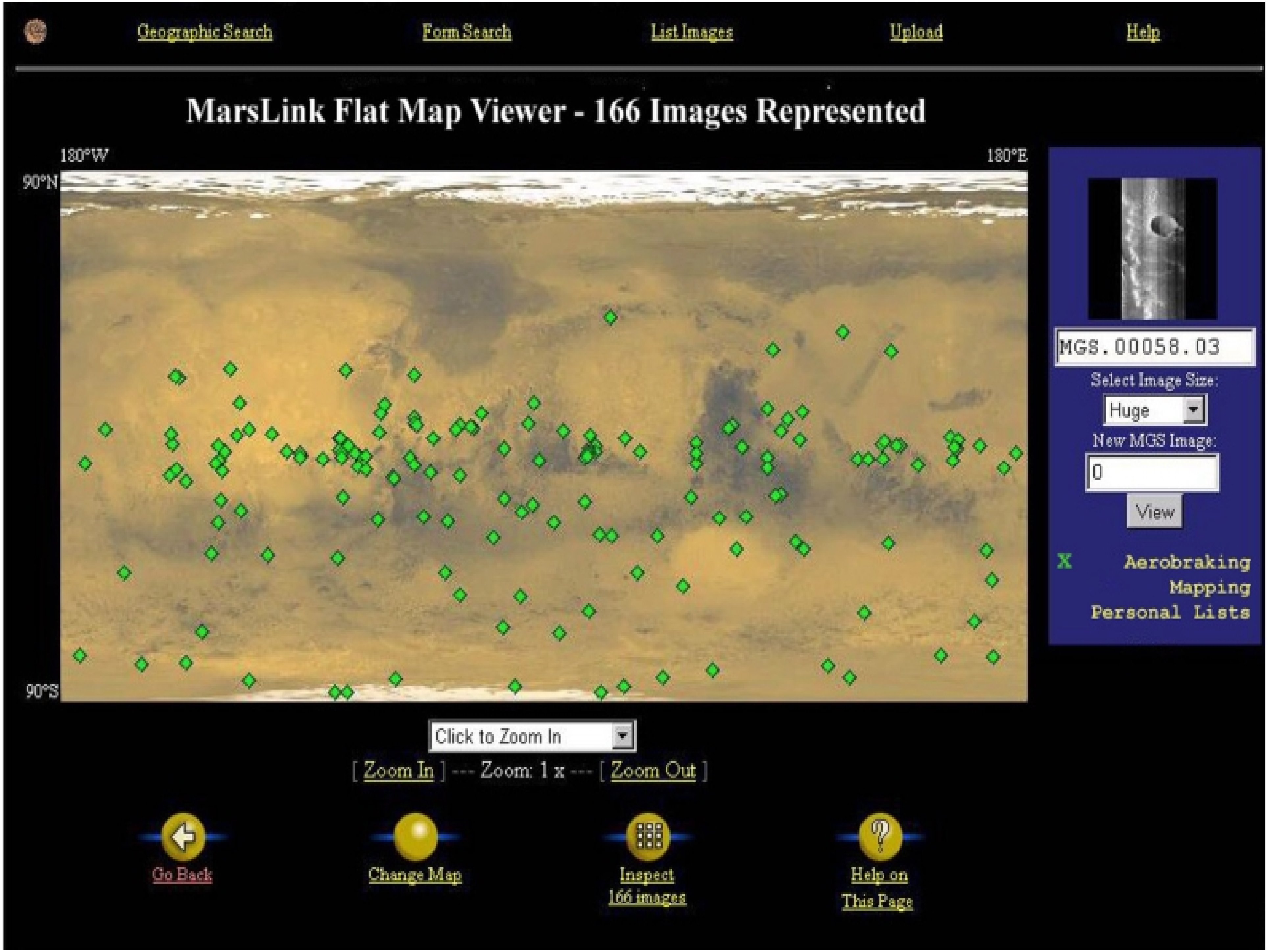
Life cycle simulation of missions to Mars in a realistic, collaborative, reusable, shareable environment. The simulation will be used to design systems and plan the scientific exploration by robots and humans of Mars and other destinations.

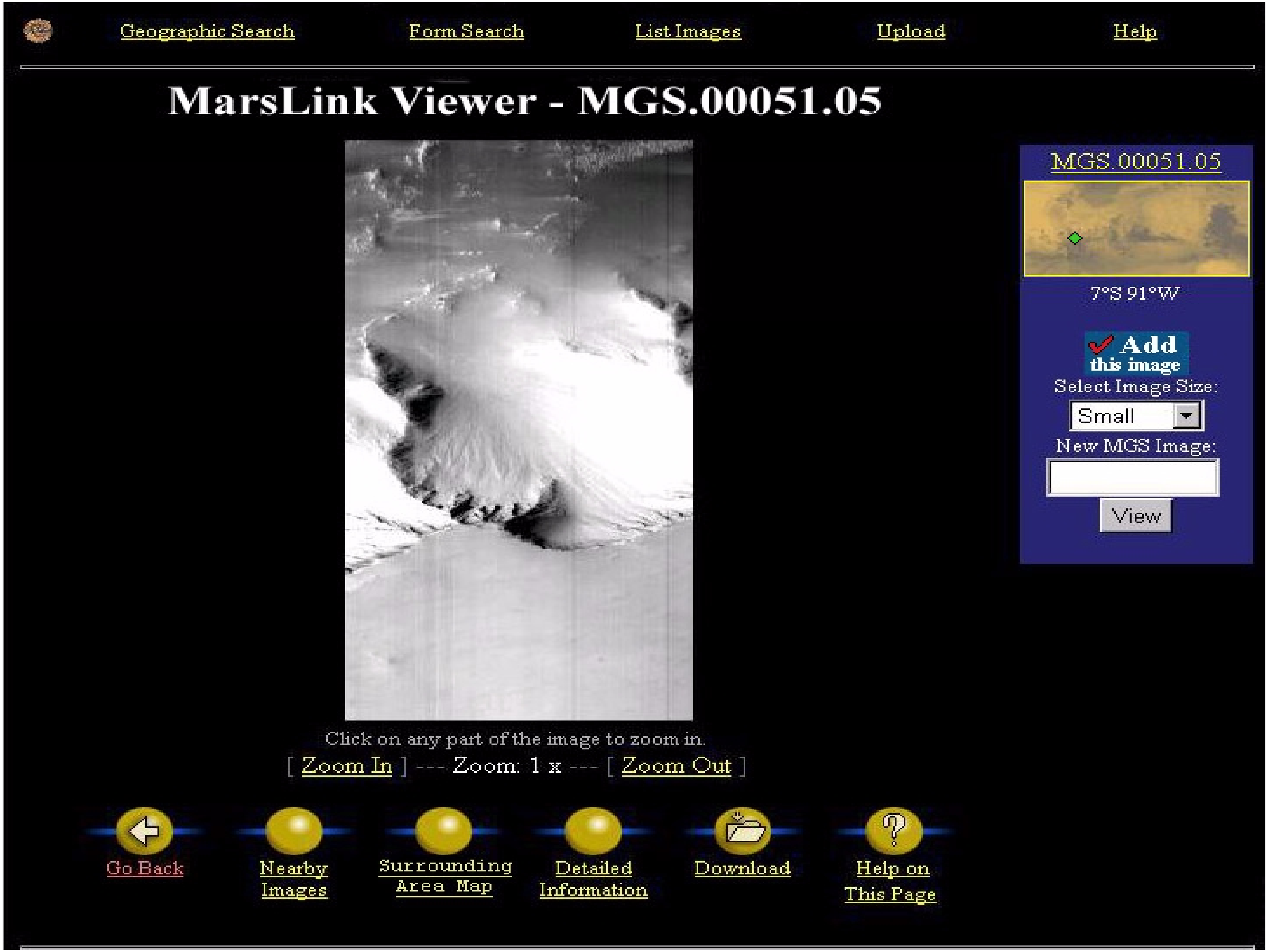


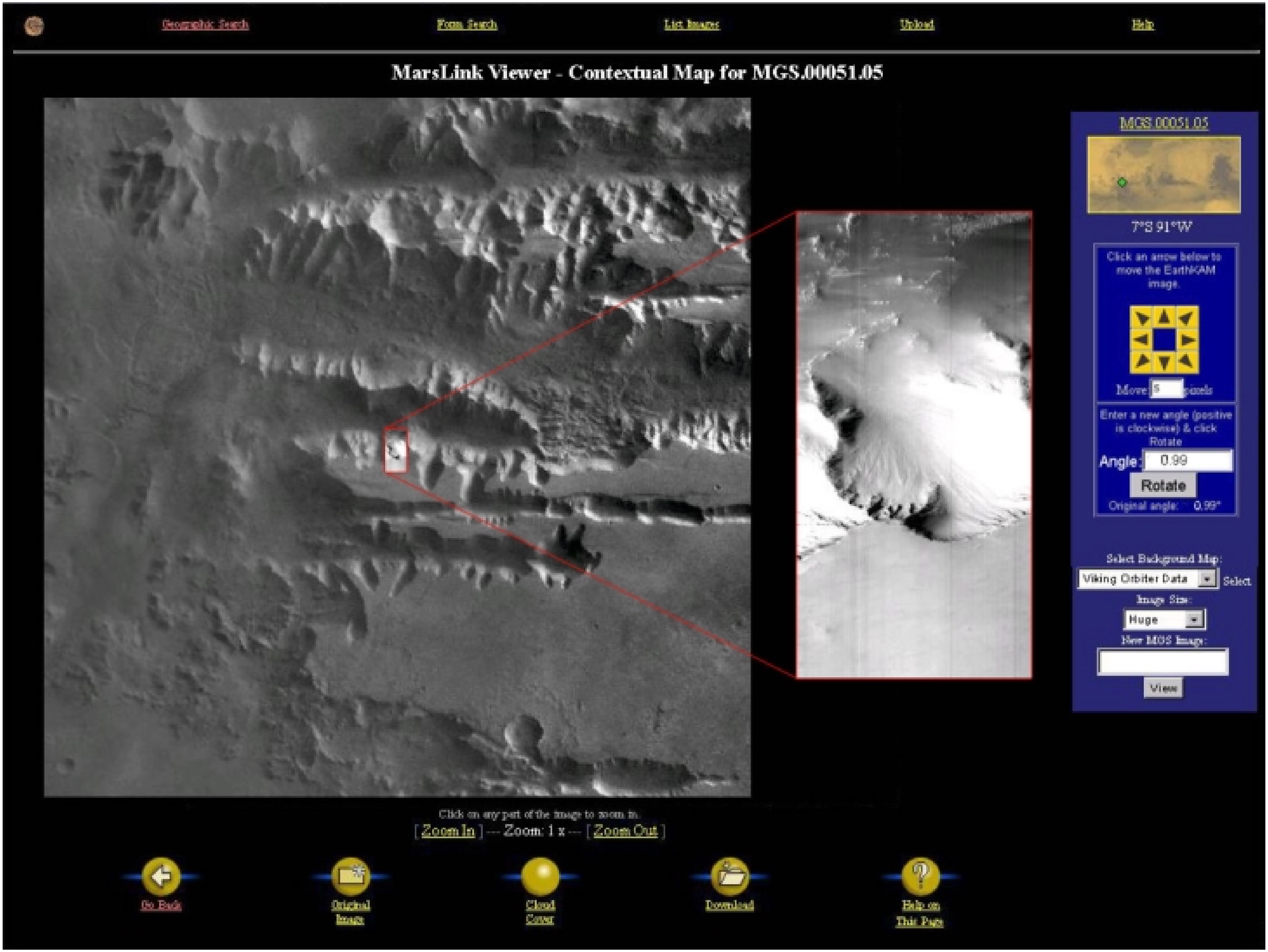
* Lead Centers

Objective(s)

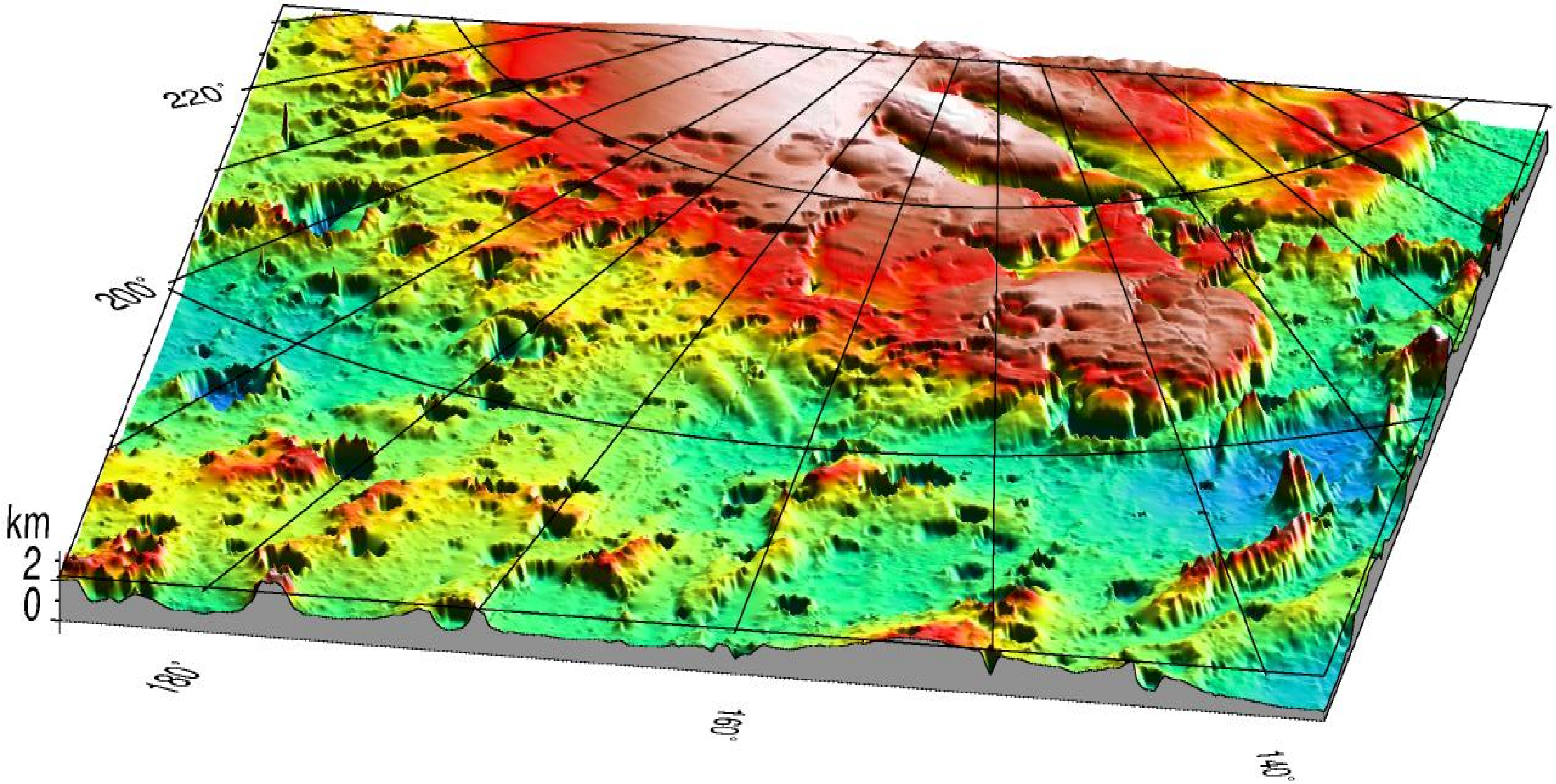
- To facilitate improved collaboration between NASA centers, contractors, academia, and partners on the Mars Program enabling:
  - optimized program and cross-enterprise architectures
  - reduced mission development time and cost
  - radical reduction in time for going from high-level strategic architecture scenario creation to detailed concept definition


Geographic Search
Form Search
List Images
Upload
Help
MarsLink Flat Map Viewer - 166 Images Represented
180°W
180°E
90°N
90°S
MGS.00058.03
Select Image Size:
Huge
New MGS Image:
0
View
X Aerobraking
Mapping
Personal Lists
Click to Zoom In
[Zoom In] --- Zoom: 1 x --- [Zoom Out]
Go Back
Change Map
Inspect
166 images
Help on
This Page

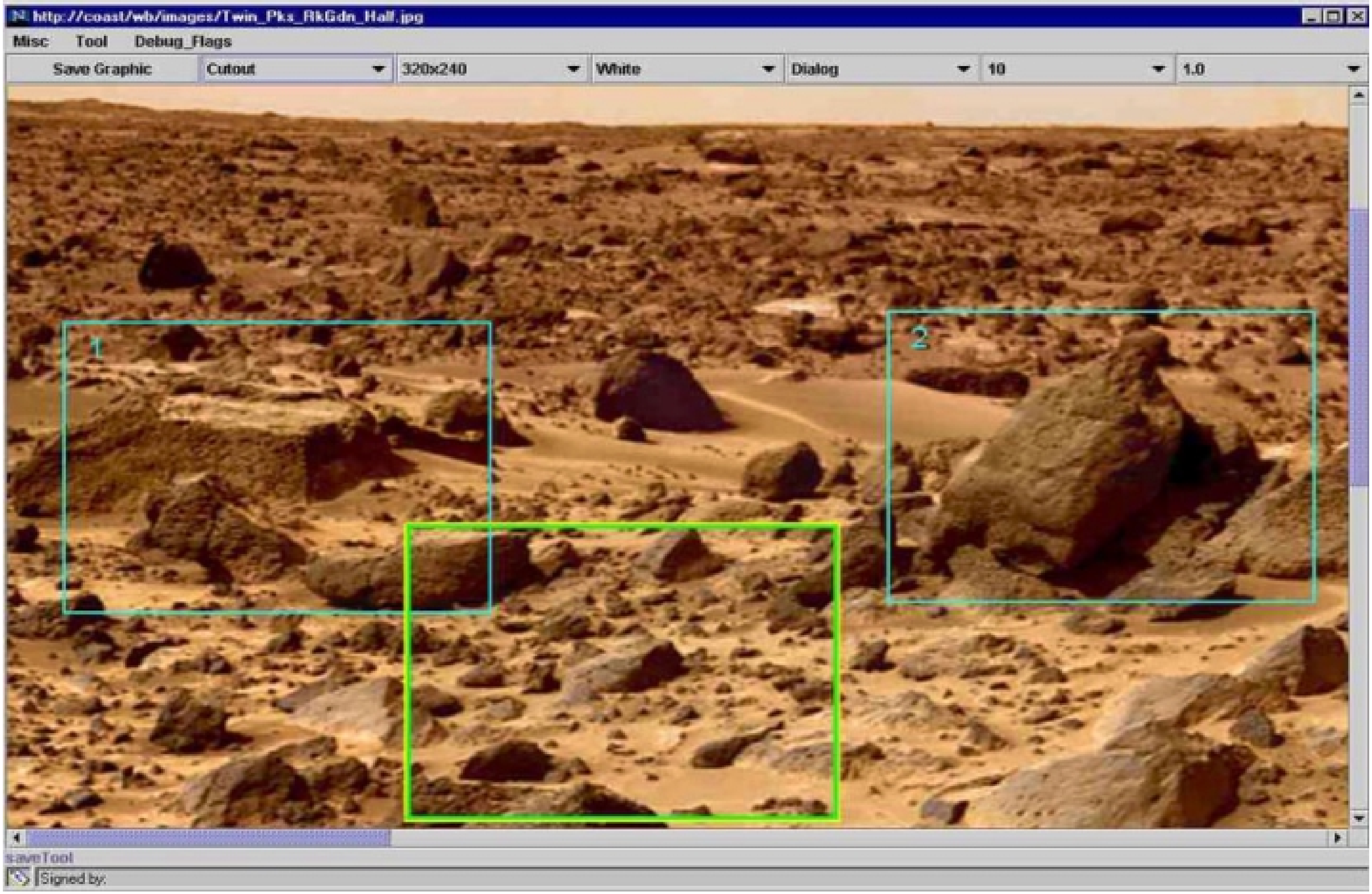

Geographic Search
Form Search
List Images
Upload
Help
MarsLink Viewer - MGS.00051.05
MGS.00051.05
7°S 91°W
Add this image
Select Image Size:
Small
New MGS Image:
View
Click on any part of the image to zoom in.
[Zoom In] --- Zoom: 1 x --- [Zoom Out]
Go Back
Nearby Images
Surrounding Area Map
Detailed Information
Download
Help on This Page


Geographic Search
Form Search
List Images
Upload
Help
MarsLink Viewer - Contextual Map for MGS.00051.05
MGS.00051.05
7°S 91°W
Click an arrow below to move the EarthKAM image.
Move: 5 pixels
Enter a new angle (positive is clockwise) & click Rotate
Angle: 0.99
Rotate
Original angle: 0.99°
Select Background Map:
Viking Orbiter Data
Select
Image Size:
Huge
New MOS Image:
View
Click on any part of the image to zoom in.
[Zoom In] --- Zoom: 1 x --- [Zoom Out]
Go Back
Original Image
Cloud Cover
Download
Help on This Page

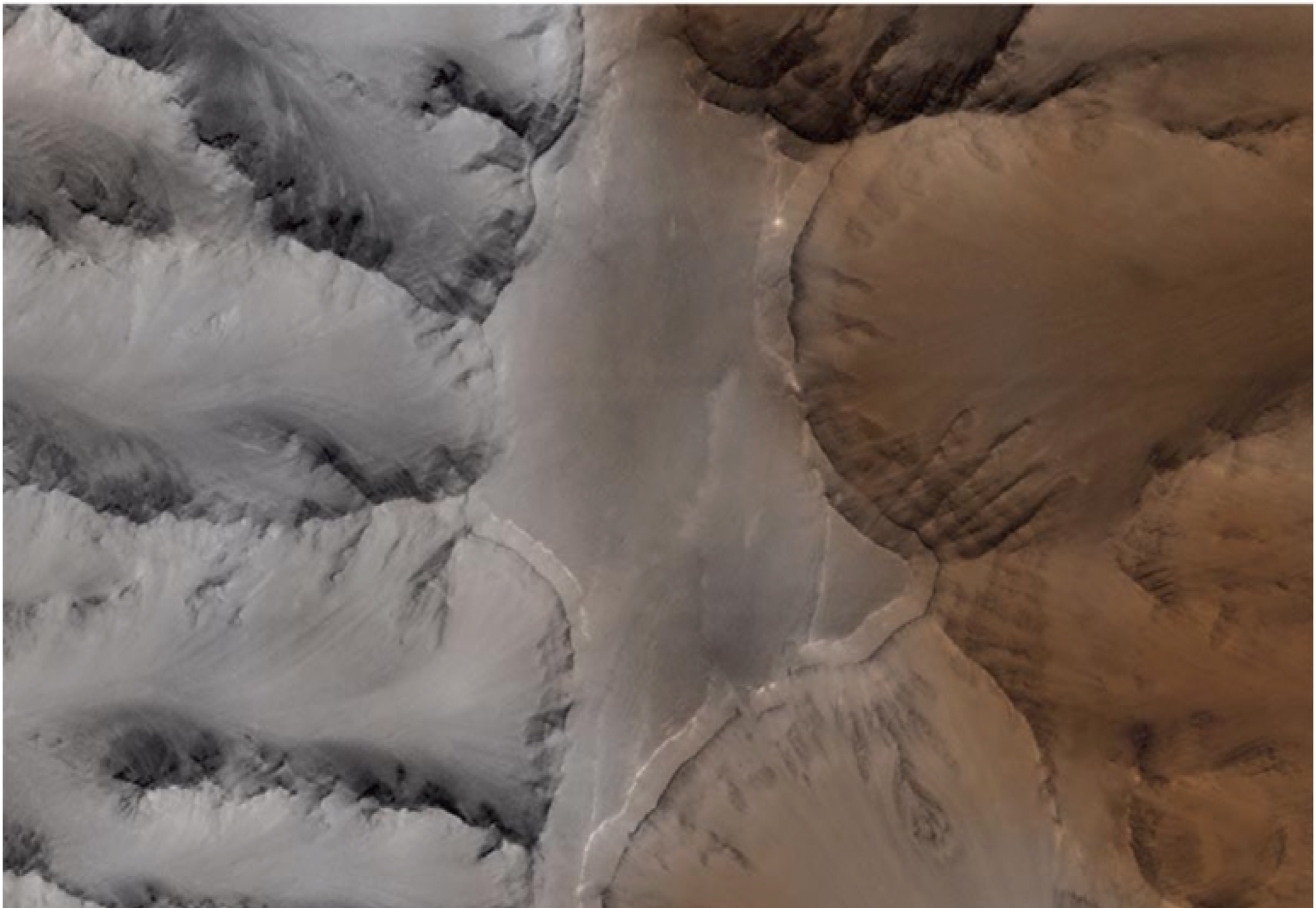
Experimenter's Notebook (MOLA Science)



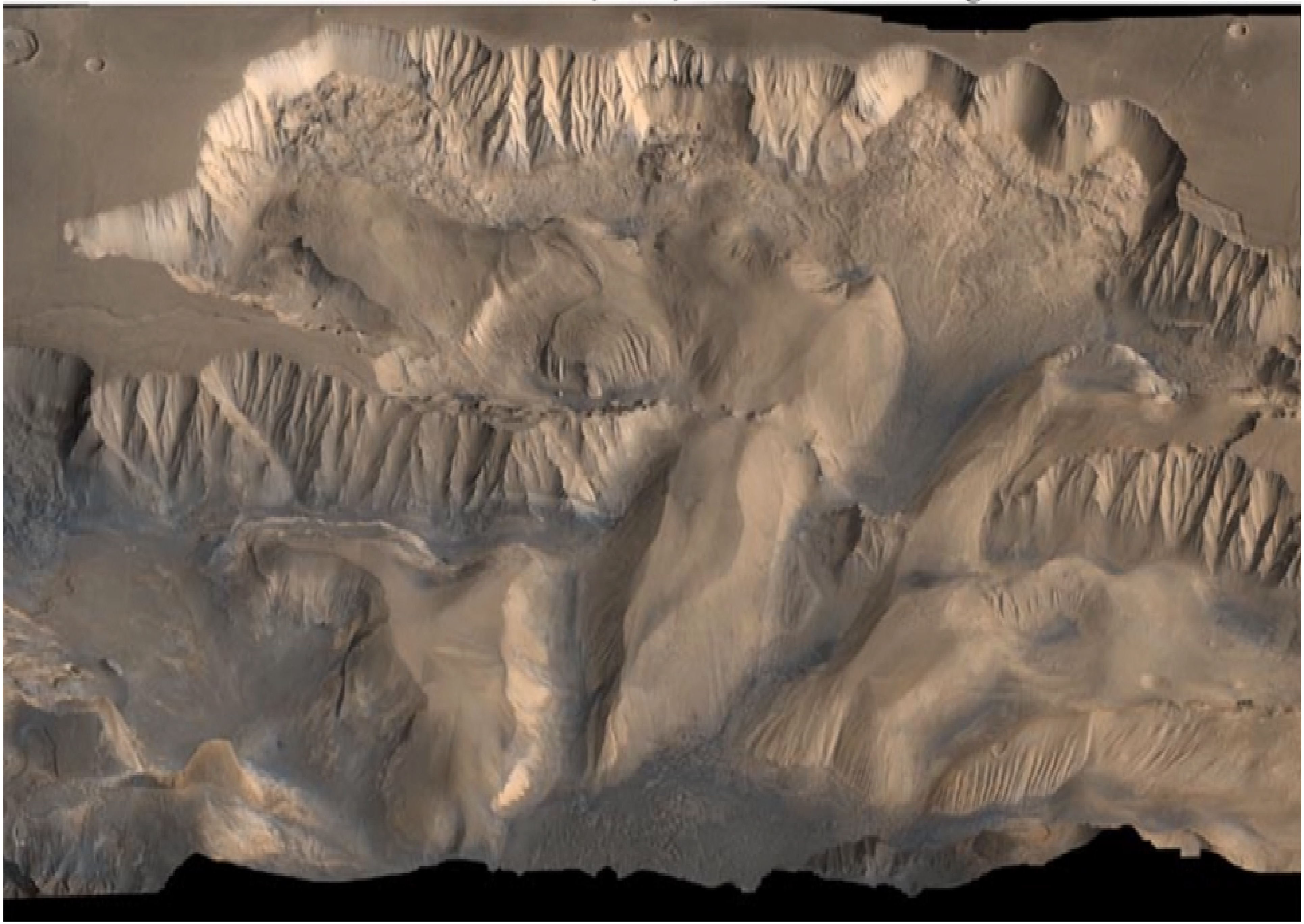
Mars White Board Mars Lander Demo
Collaborative Science Environment



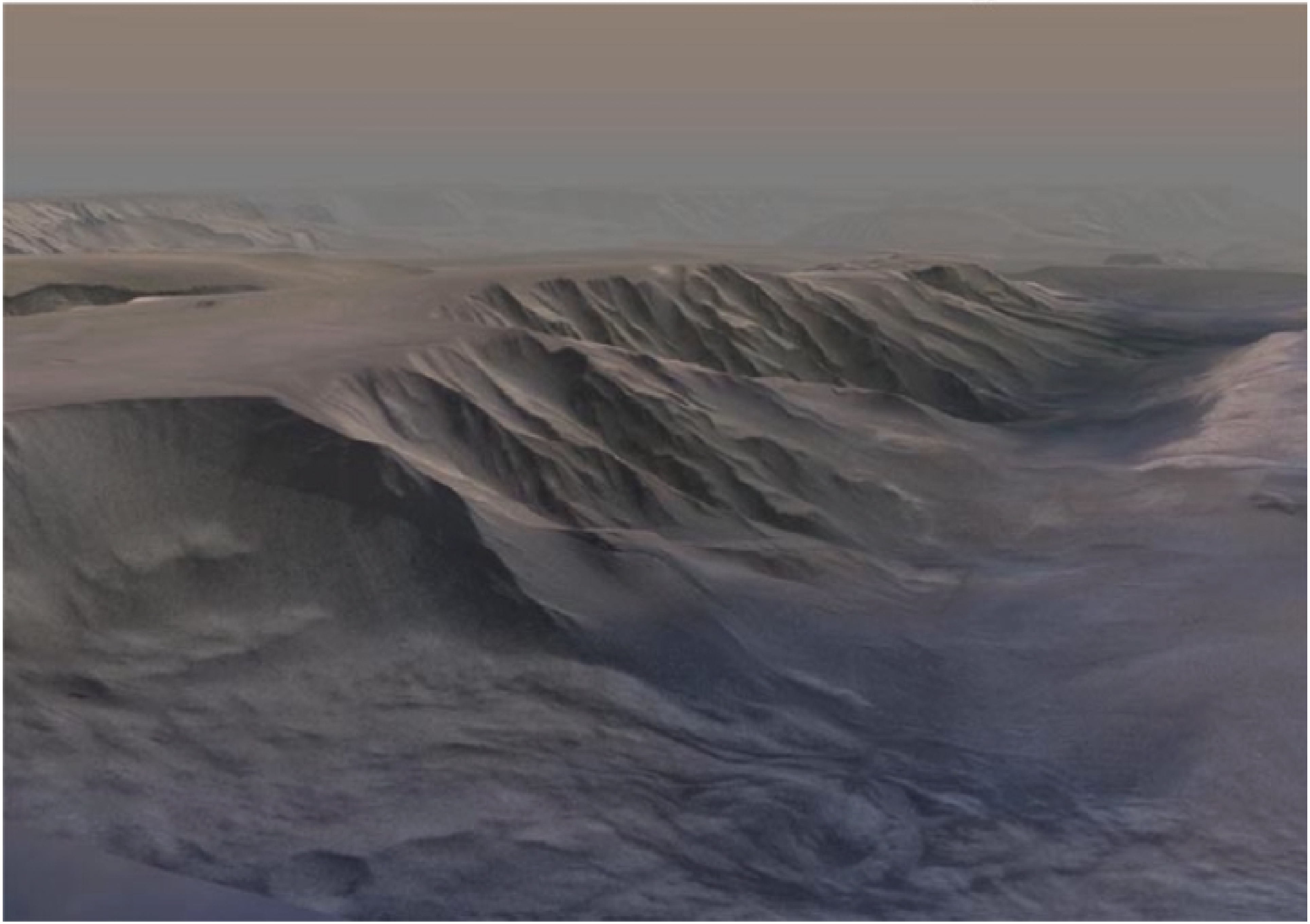
Visualization of Mission Operations Mars Orbiter Demonstration (MOD)



Global Virtual Mars (GVM) Virtual Mission Design



Mars Environment Distributed Immersive Research Technology DIRT/WINDS



“A Historic Opportunity…”

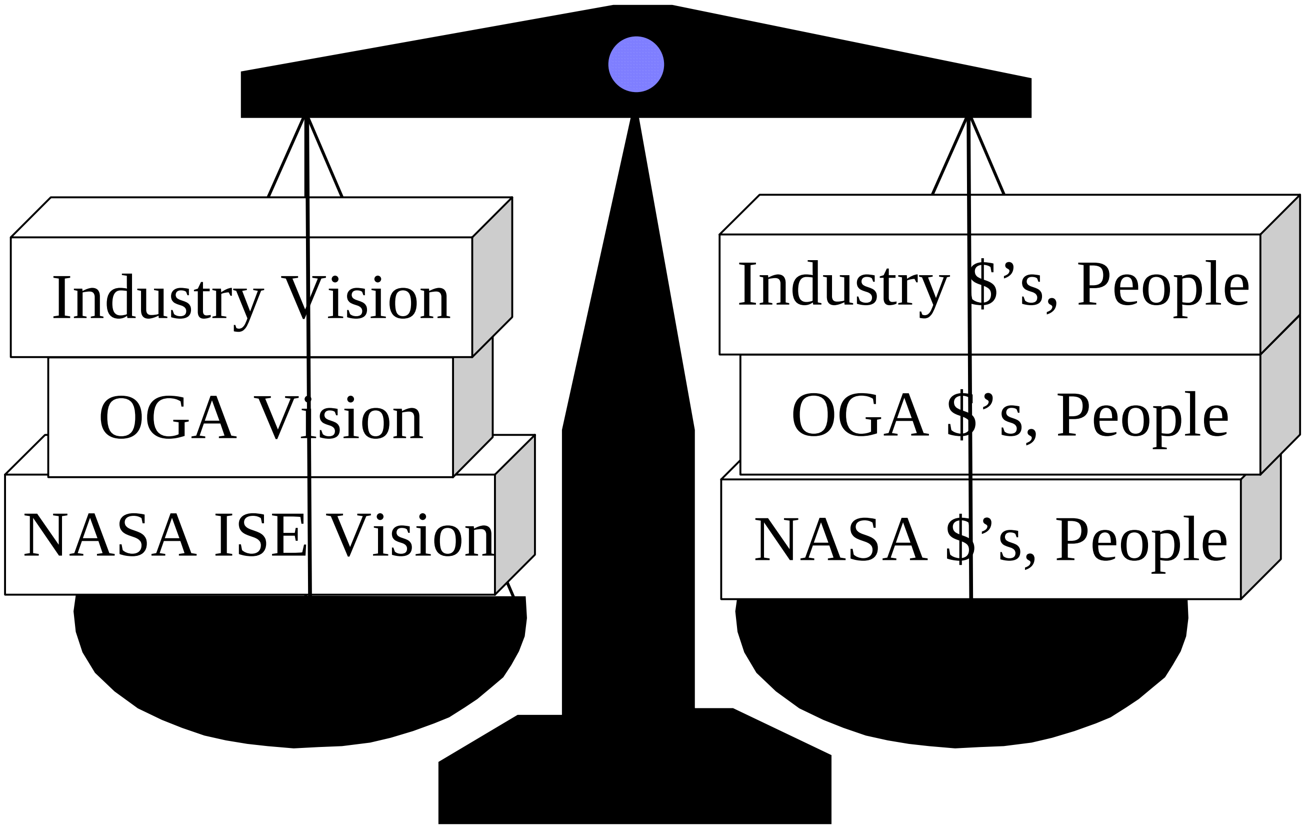
“A historic opportunity now exists to develop AEE technologies and systems that could revolutionize computer-based engineering processes, just as the Internet has revolutionized computer-based communications. This opportunity is too big for any one organization to realize on its own.”

“Advanced Engineering Environments”

National Research Council Phase 1 Study - Page 34

Note: Phase 1 Study Report can be found at: http://www.nap.edu/catalog/9597.html

ISE Will Seek Agreements with Related-Activities



Appropriate leveraging and cooperative research can produce <u>balance</u> between needs and resources for participating organizations/programs

2nd Annual GSFC-JPL Quality Mission Software Workshop

Future Infrastructure and Technology for Software Development

Automated Software Verification

Information Sciences & Technology

Ames Research Center

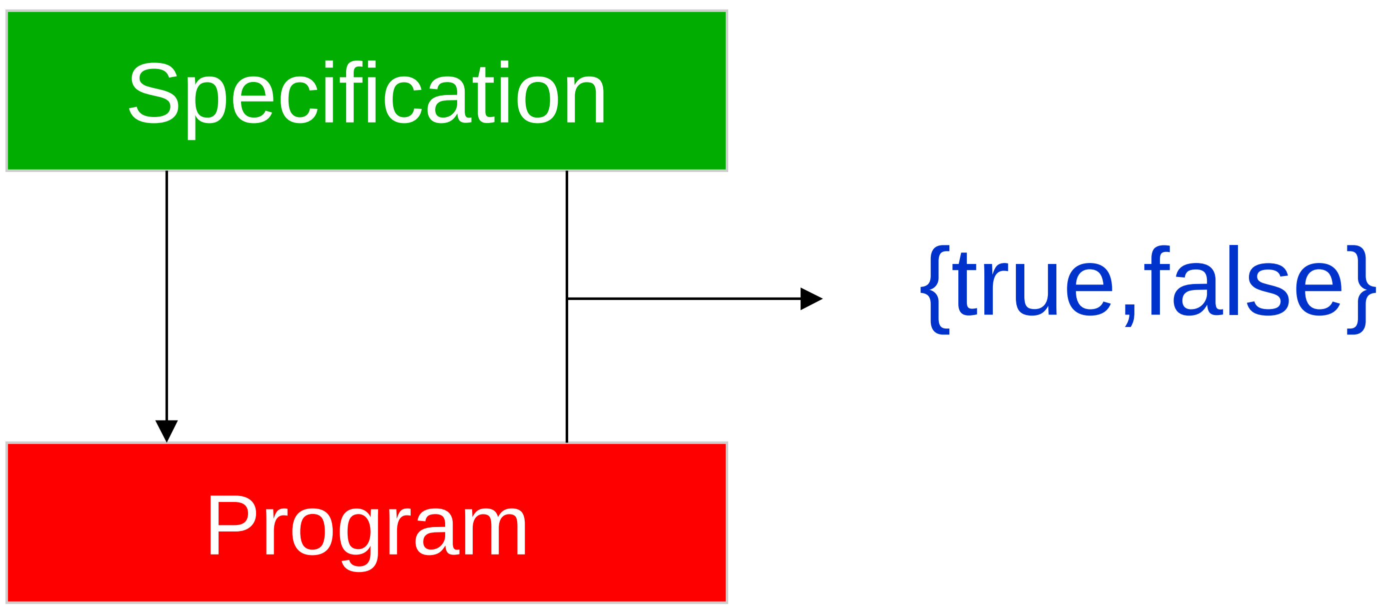
NASA

Klaus Havelund

May 16, 2000

Automated Software Engineering

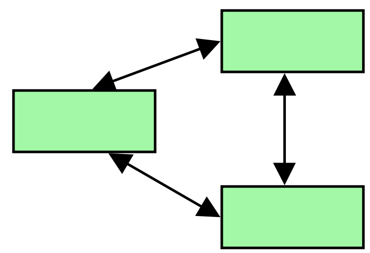
- Group created in 1997 with two subgroups:
- Synthesis : specification -> program
- Verification : program * specification -> {true,false}



Problem: Multi Threaded Systems

Sequential Systems : testing "works"

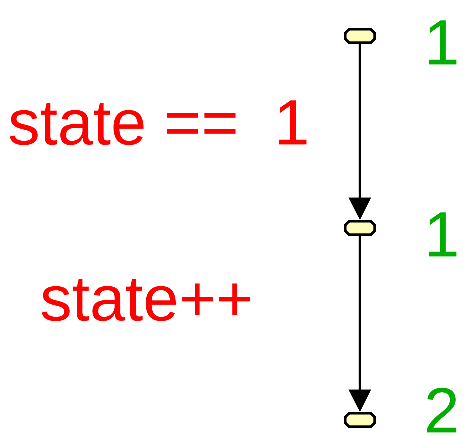
Multithreaded Systems : testing becomes hard

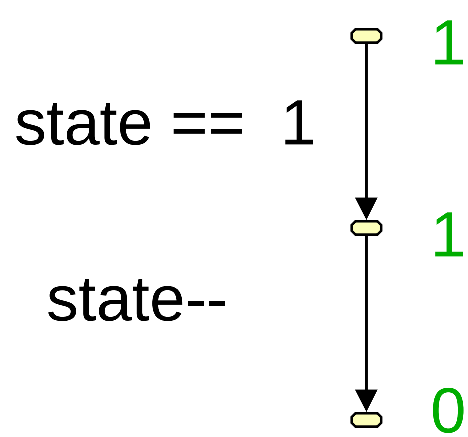


Two Sequential Programs

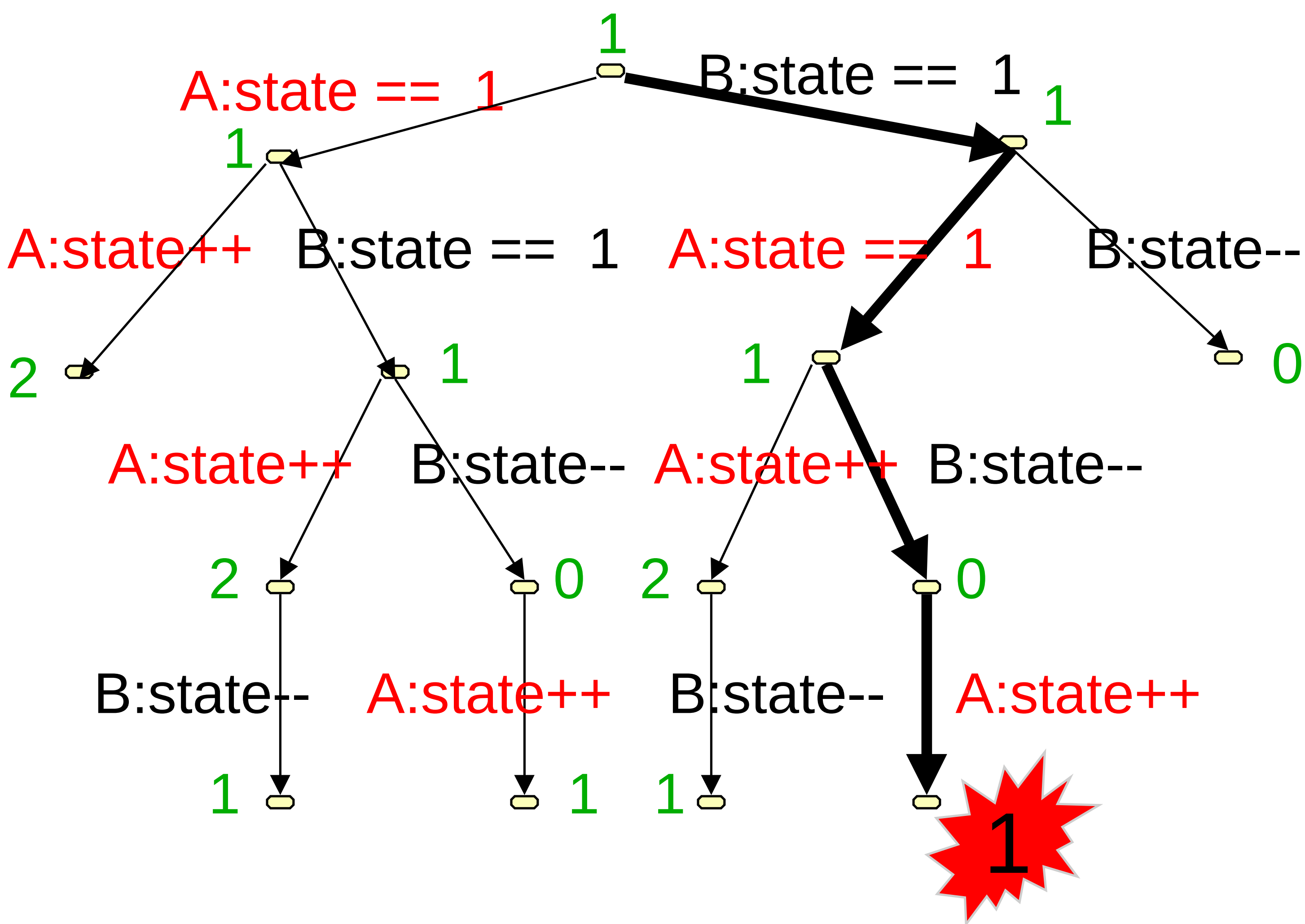
A:	**B:**
if state == 1 then	if state == 1 then
state++;	state--;
assert(x == 2);	assert(x == 0);

Initial state == 1





The Concurrency Syndrome



The Concurrency Test Problem

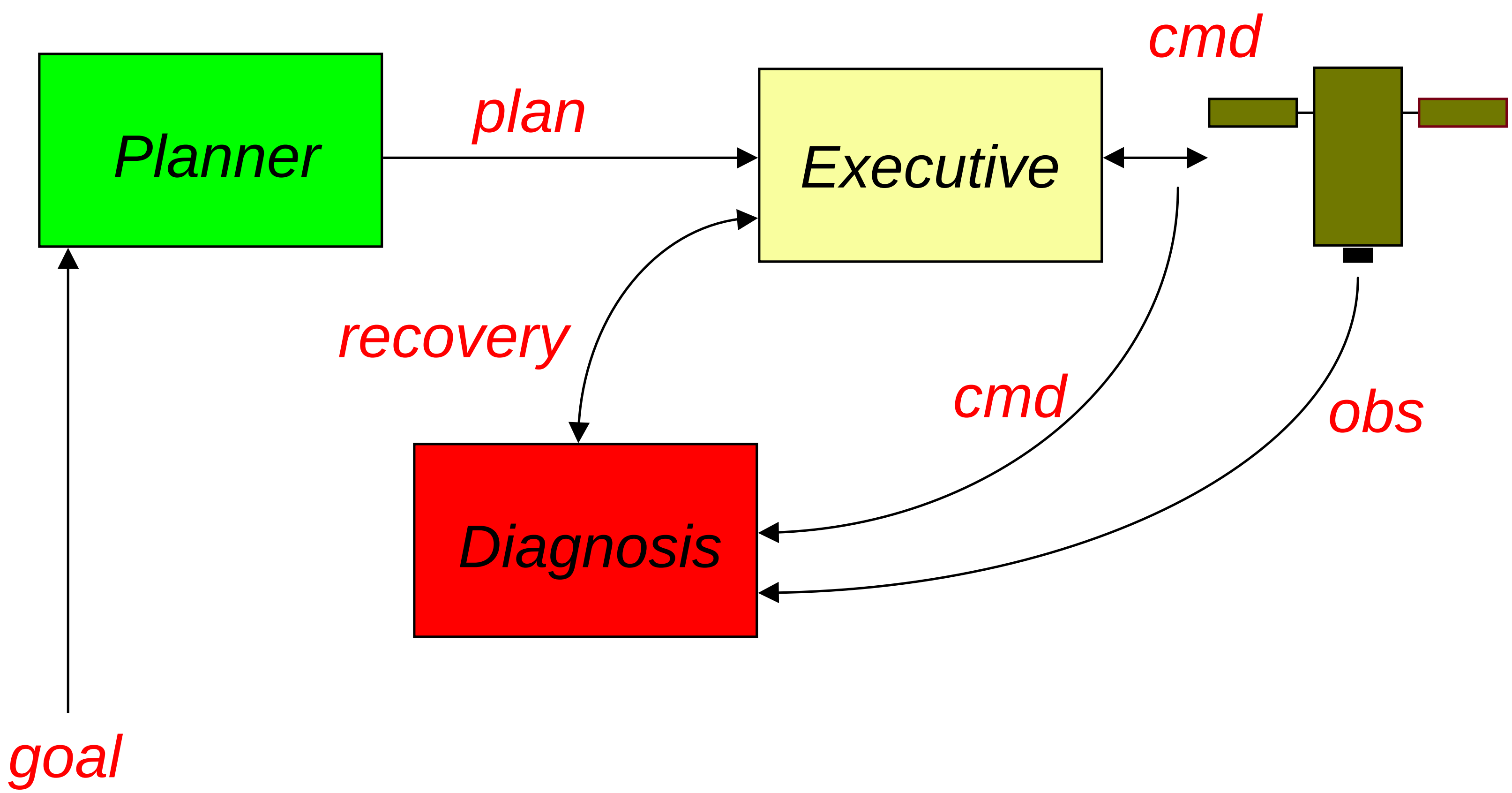
- Not enough to control test inputs
- Also requires control over **scheduler**
- Even worse : scheduler may differ between platforms

A Major Case Study: The Remote Agent

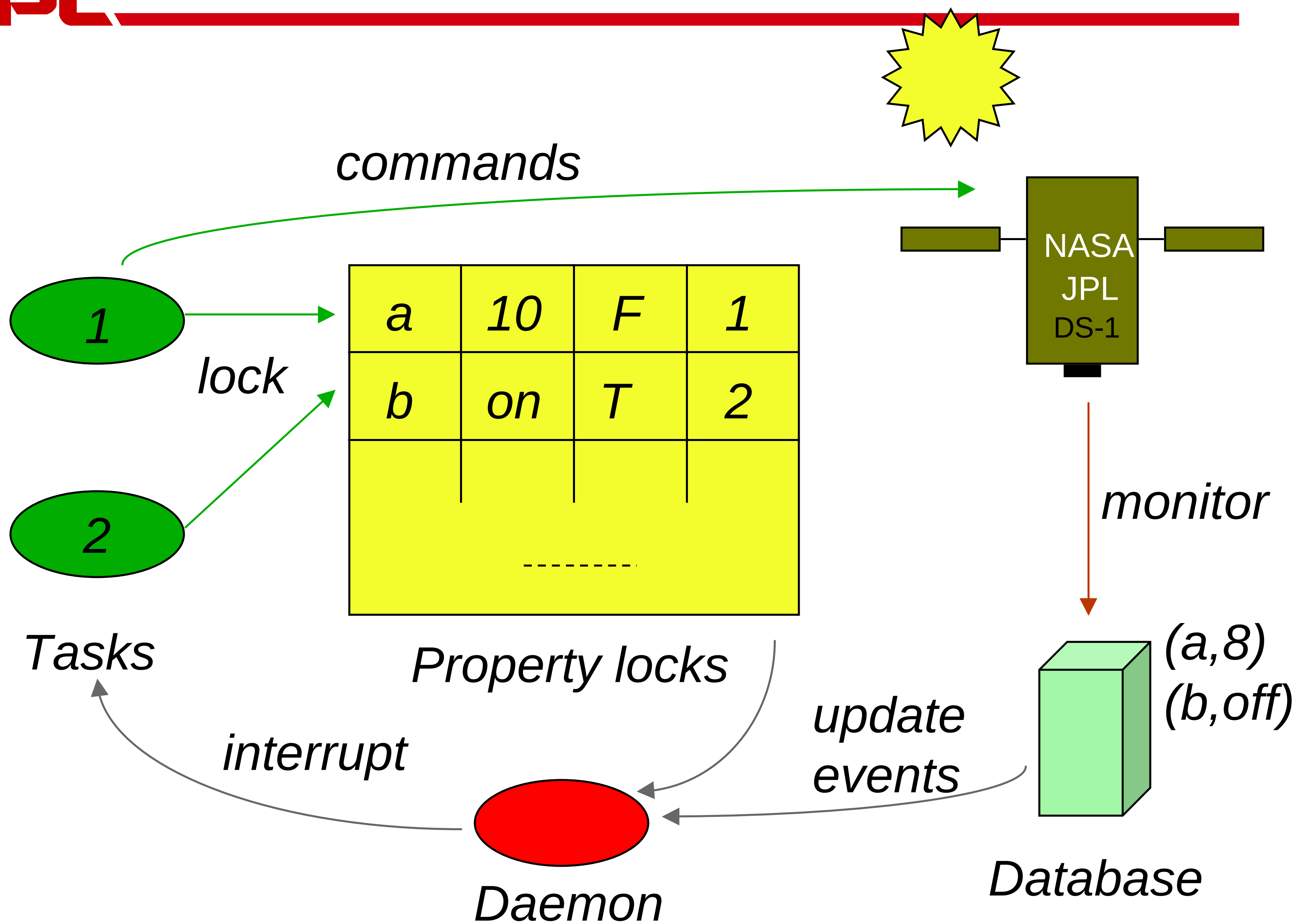
An AI based controller for

Deep Space 1

The Remote Agent



Informal Description of Exec.



Analyzed using SPIN

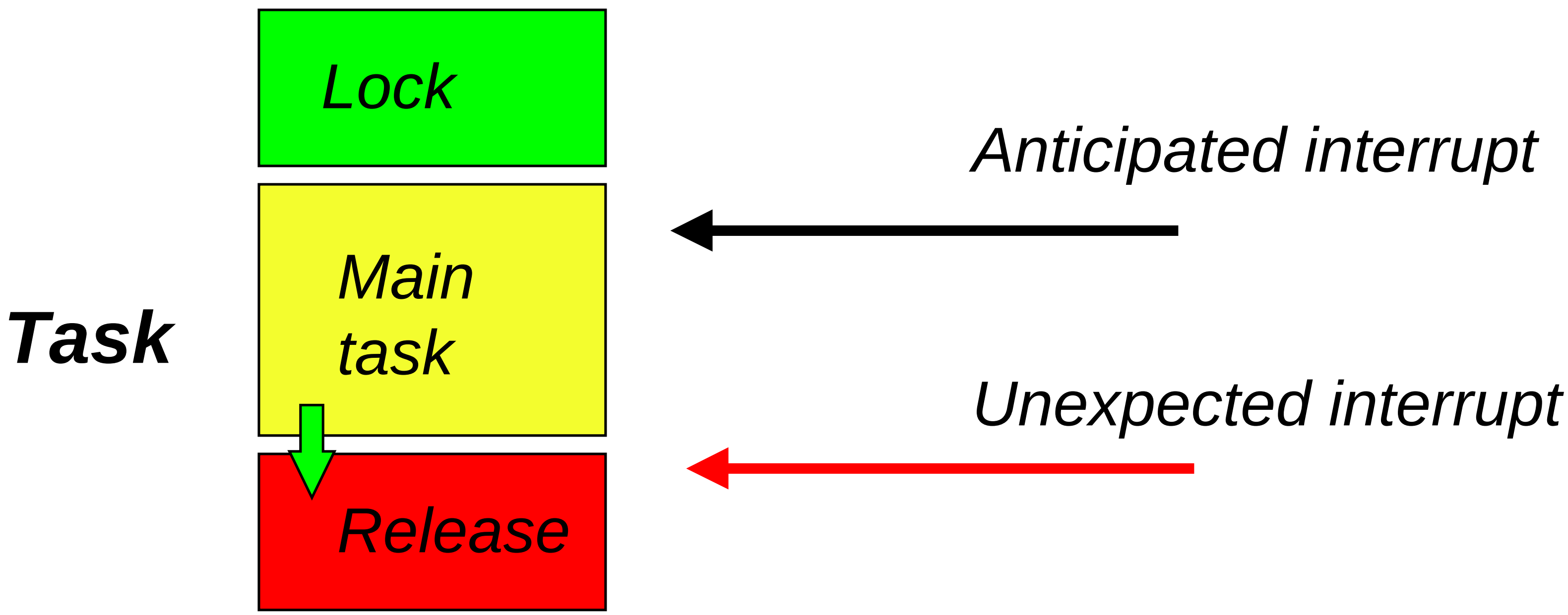
```
inline achieve_lock_property(this,p,err){
  TaskId owner;
  find_owner(p,owner);
  if
  :: owner == this ->
        achieve(p,err);
        locks[p.name].achieved = true
  :: else ->
        wait_for_event(this,MEMORY_EVENT,p)
  fi
}
```

Properties Stated by Engineers

Release Property:
A task releases all its locks before it terminates.

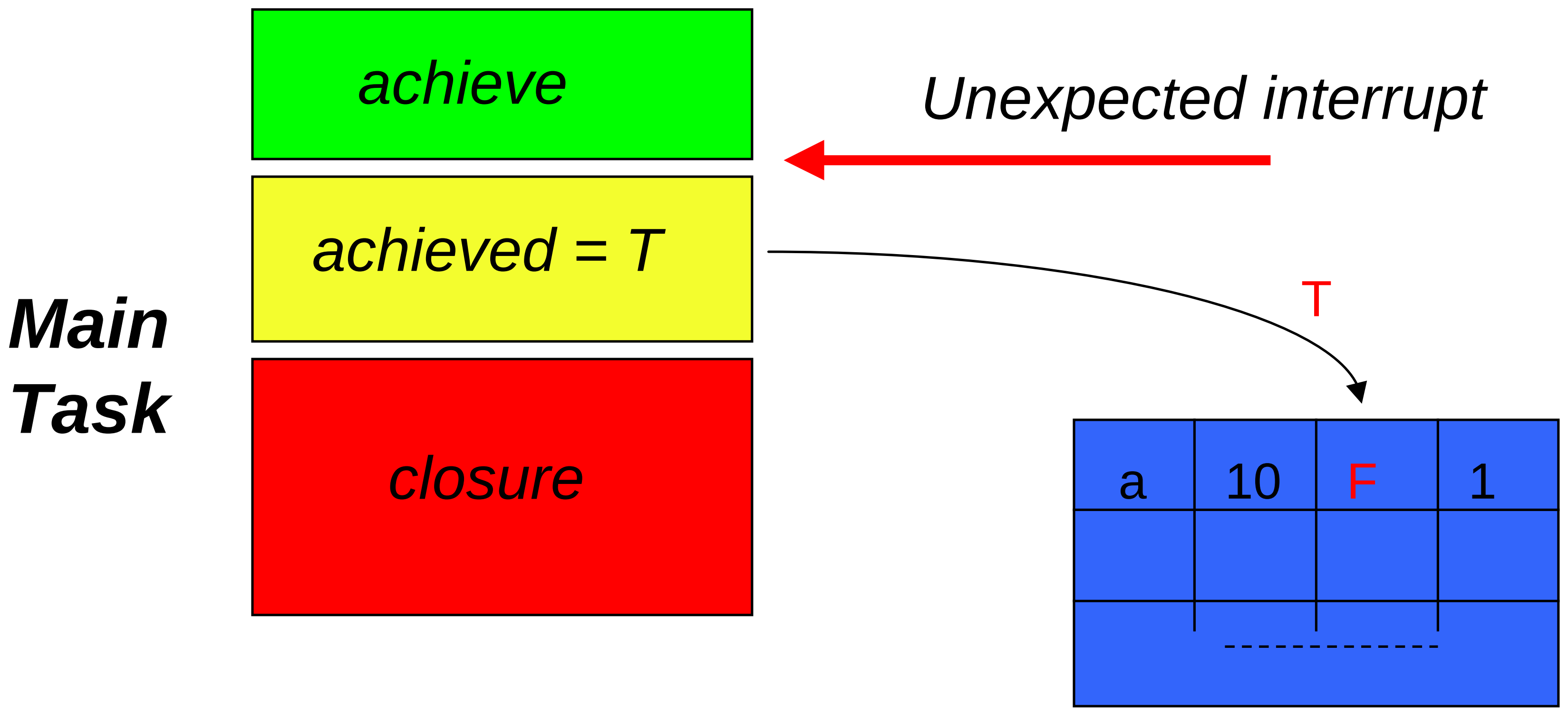
Abort Property:
If an inconsistency occurs between the database and an entry in the lock table, then all tasks that rely on the lock will be terminated, either by themselves of by the daemon.

Report 1 : Lock Releasing



= *Unwind-protect in LISP* (**finally** *in Java)*

Report 2 : Achieve Procedure



Report 3 : Check Locks Procedure

- no change discovered
- no interrupts

Check Locks

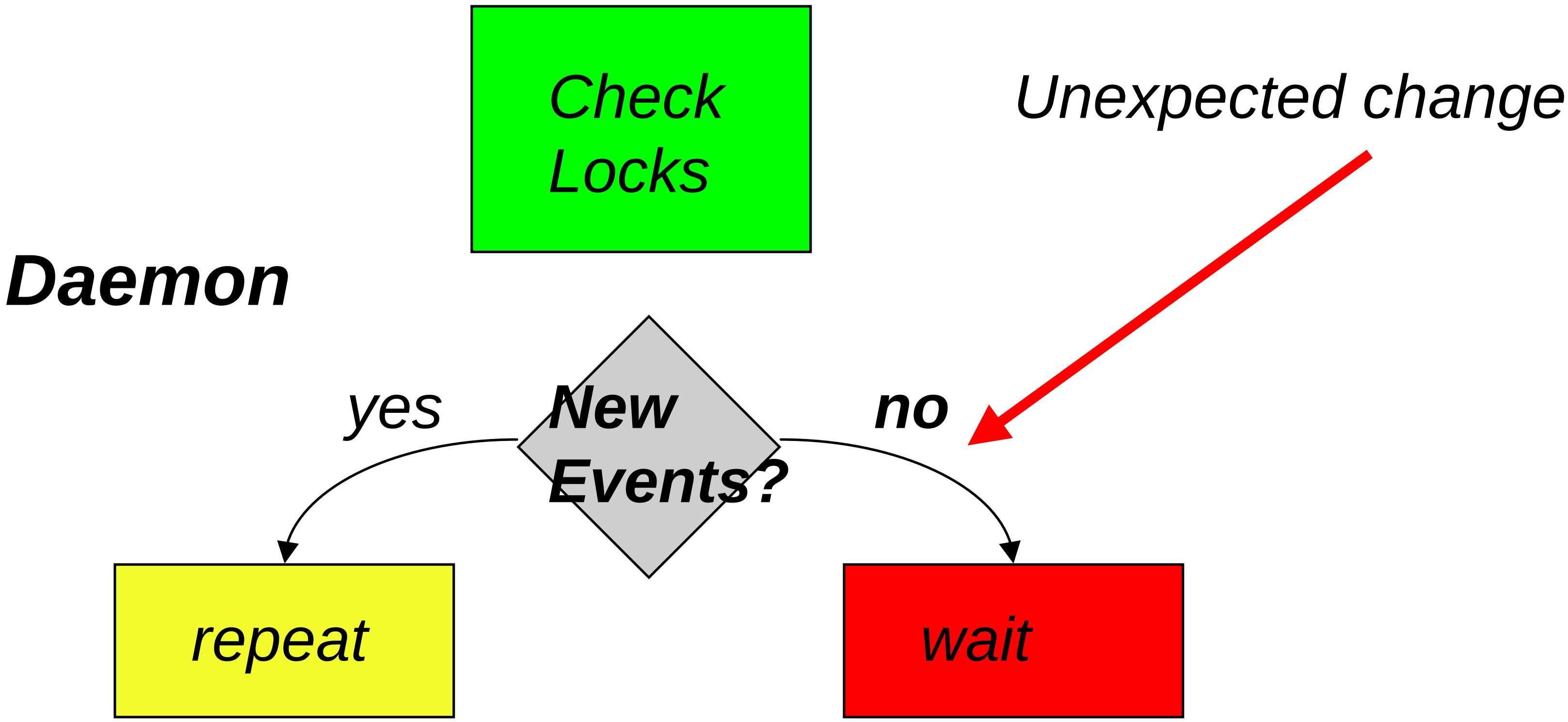
Check locks :
<u>interrupt</u>

Unexpected change

Check locks :
<u>Recover rest</u>

- change discovered
- automatic recovery
- **but NO interrupts**

Report 4 : Waiting Procedure

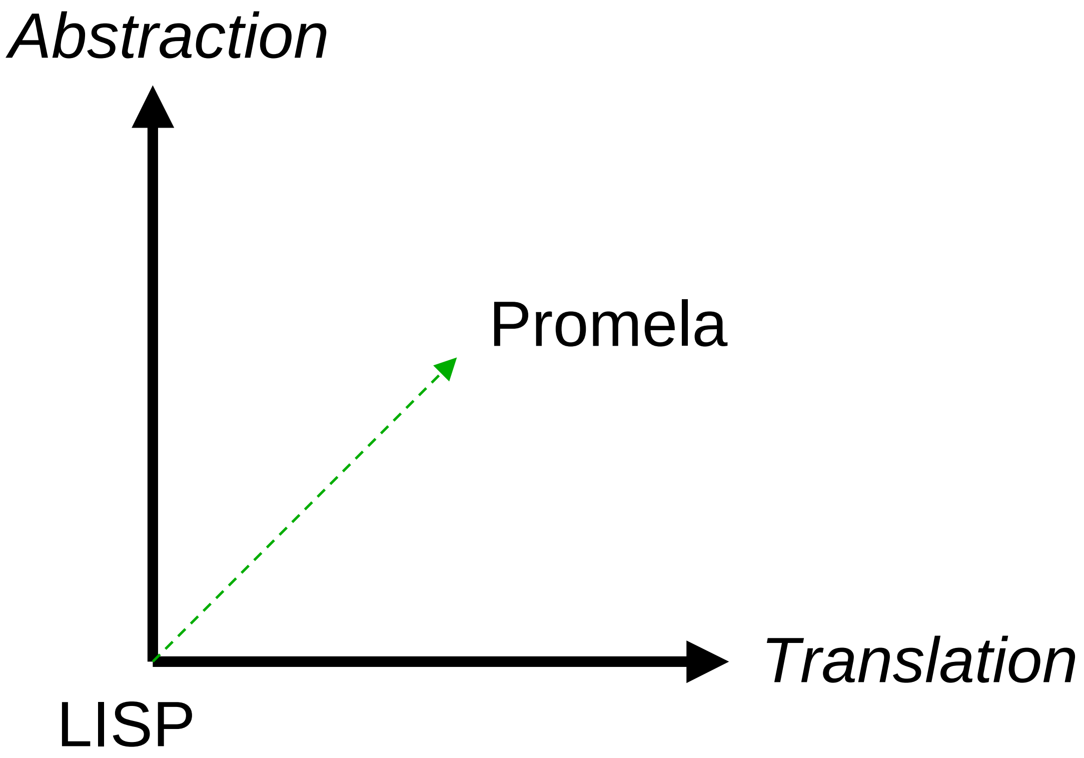


Did Our Work Have Any Impact?

“You’ve found a number of bugs that I am fairly confident would not have been found otherwise. One of the bugs revealed a major design flaw (which has not been resolved yet). So I’d say you have had a substantial impact. If nothing else you have helped us improve the quality of our product well beyond what we otherwise would have produced.”

Programmer

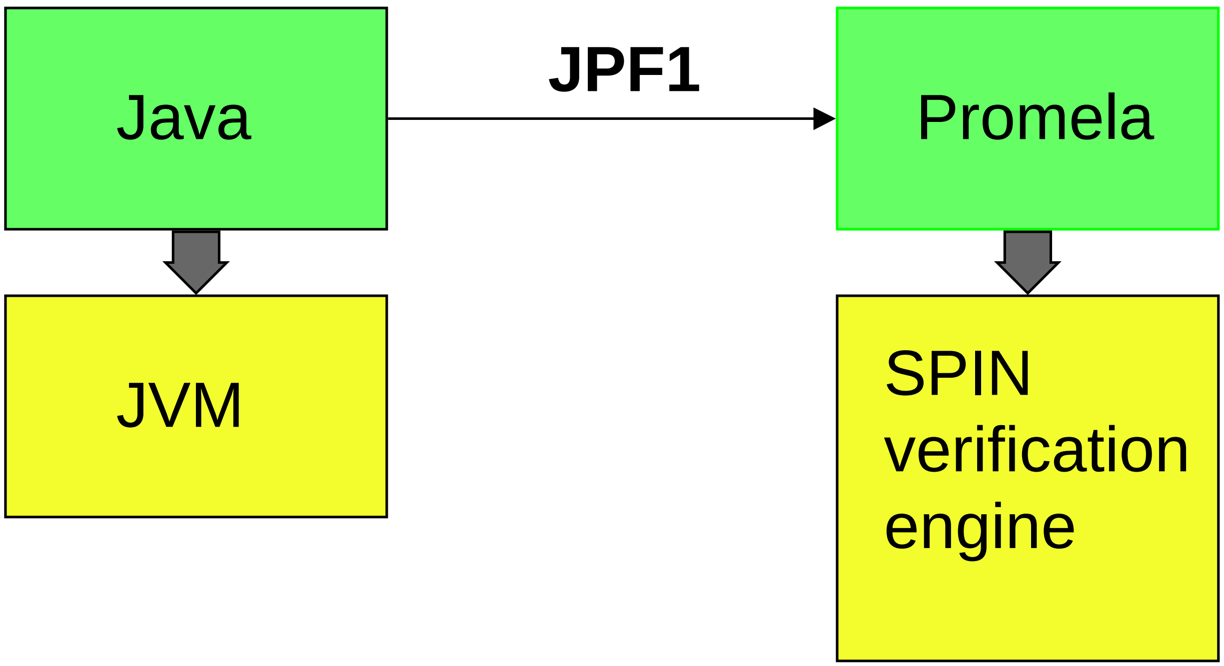
Lessons Learned



Java PathFinder 1

A Translator from Java to SPIN

JPF : Java -> SPIN (Promela)



Java PathFinder 2

A Byte Code Model Checker

JPF2 = New JVM + MC

- Written in Java ("Java in Java")
- New java virtual machine designed for verification
- Also designed for modification (modular, highly OO)
- Handles libraries well. Possible to cover 100% of Java
- Generic : can in principle be instantiated with another virtual machine (UML statechart VM being developed)

Characteristics of JPF2

- Finds deadlocks, assertion and invariant violations
- All interleavings are explored.
- States are stored in hash table.
- Revisit of a state causes backtracking.
- Partial orders reduce traces examined.
- Error trace is printed for errors detected.

Example

```
class W extends Thread{
  Monitor monitor;

  public void run(){
    monitor.Wait();
  }
}
```

```
class S extends Thread{
  Monitor monitor;

  public void run(){
    monitor.Signal();
  }
}
```

Example

W

Public void run()
0: aload_0
1: getfield W.monitor
4: invokevirtual Monitor.Wait
7: return

S

Public void run()
0: aload_0
1: getfield S.monitor
4: invokevirtual Monitor.Signal ()
7: return

Dealing with Big State Spaces

Testing and Abstraction

- Problem: exploring all interactions is often intractable due to the large or often infinite state space of software.
- Solution 1: dial towards TEST, and explore only some traces.
- Solution 2: abstract system to tractable size by ignoring details, and then explore all traces.

Runtime Analysis – General Idea

- Run the program once (one trace!).
- Information is collected about the programs behavior.
- The information is analyzed, and potential problems are identified, for example possible race conditions and lock order conflicts.
- Information can be analyzed on the fly or after the run.
- Potential problems result in warnings issued and the programmer will check whether they may lead to problems or not.
- Note that problems do not have to occur in order to be detected! This is the strength of the method.

The Eraser Algorithm

- Detects potential data races.
- A data race is when two threads access a shared variable, at least one access is a write, and no mechanism is used to prevent simultaneous access.
- Can be caused by a missing critical section.
- RAX deadlock was caused by missing crit section.

Example

Thread1

```
old_count := E.count;
while (true){
  if (old_count == E.count)
     E.wait();
  old_count := E.count;
  /* do job */
}
```

Thread2

```
...
E.signal()
```

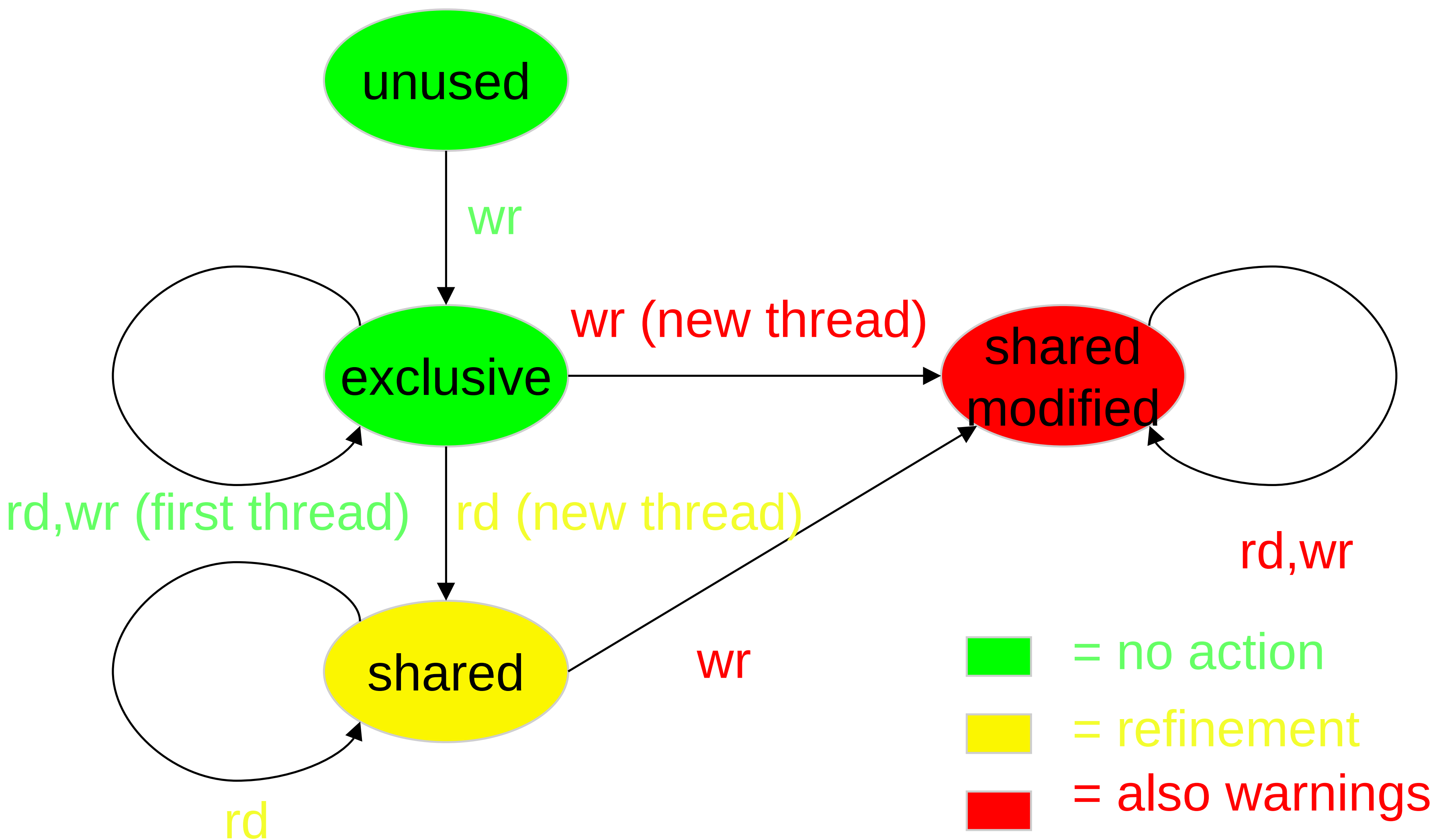
Event

```
Queue q;
int count;

void wait(){
  add to queue;
  block
}

void signal(){
  empty queue;
  count++
}
```

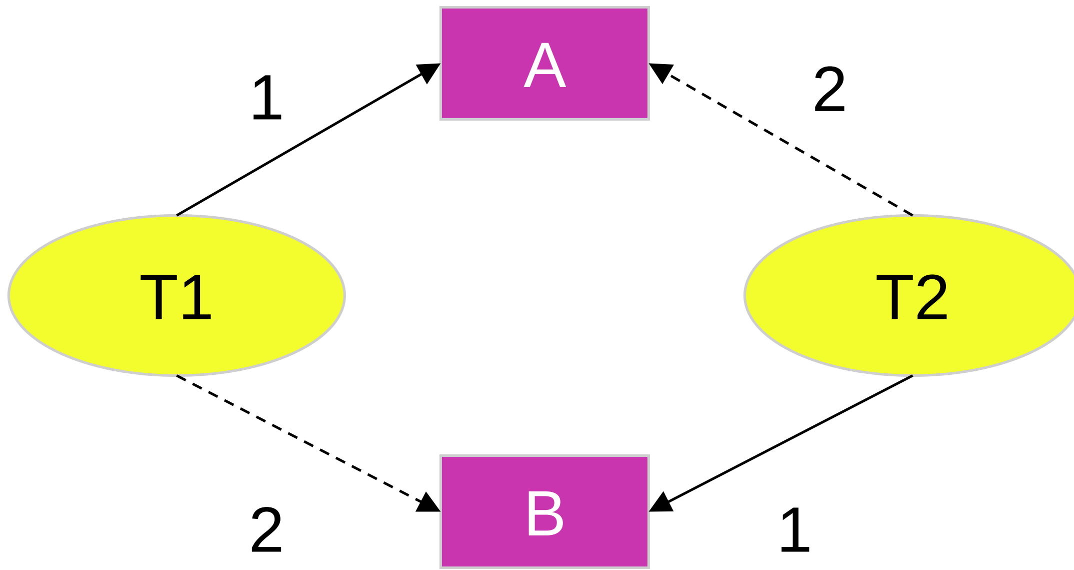
The Basic Algorithm

		Locks held by thread	**Lock set of variable**
		{}	null
T1	synchronized(lock1){	{lock1}	
	v = v+1		{lock1}
	}		
		{}	
T2	synchronized(lock2){	{lock2}	
	v = v+1		{}
	}		

The Extended Algorithm



Deadlocks Due to "Lack of Order"



Problem: T1 locks A first
T2 locks B first

There is <u>no order</u> on the locks, such as for example: A < B

Goddard Space Flight Center JPL

Example for Illustrating Algorithm

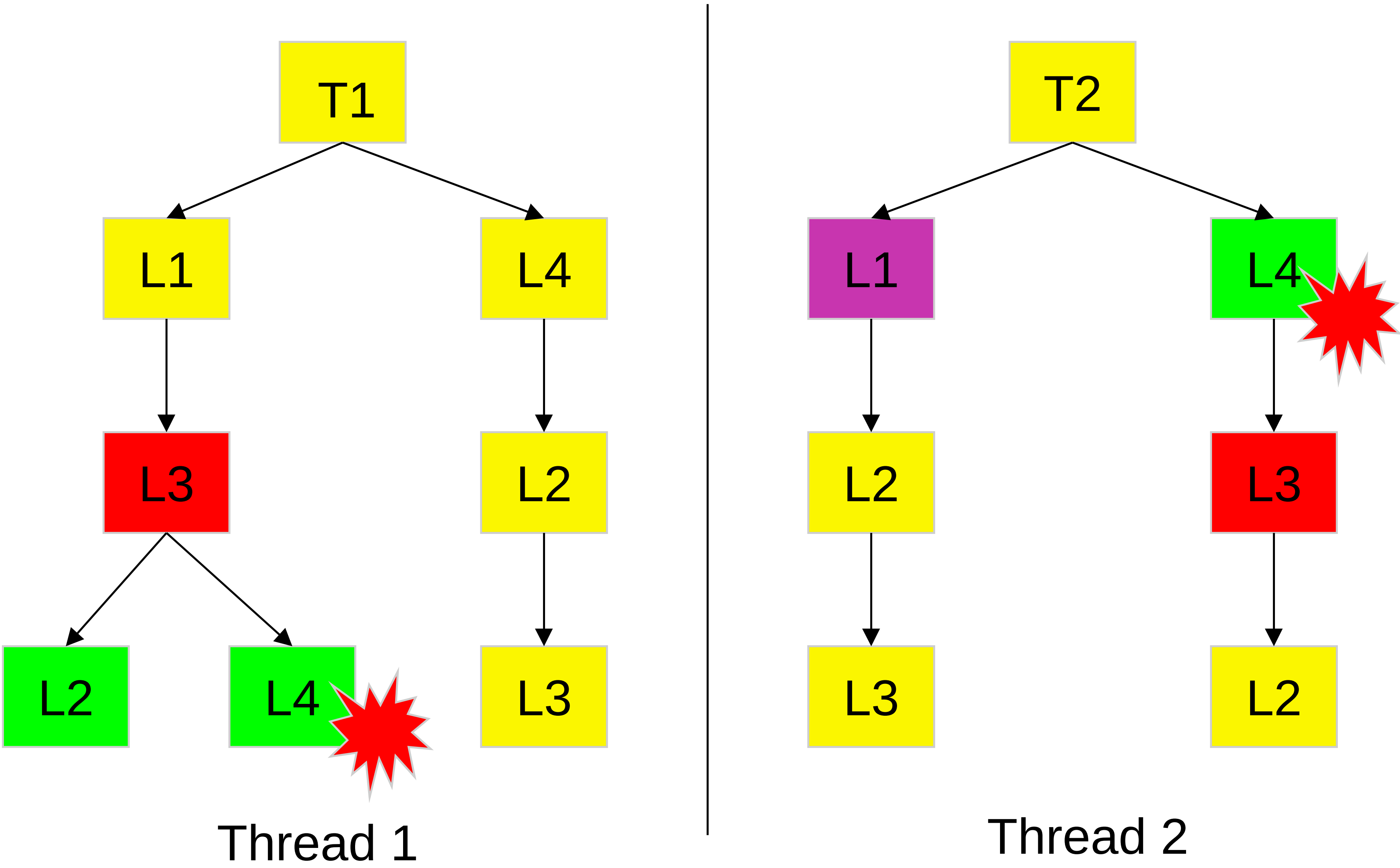
Thread 1:

```
synchronized(L1){
  synchronized(L3){
    synchronized(L2){
      …
    };
    synchronized(L4){
      …
    }
  }
}
```

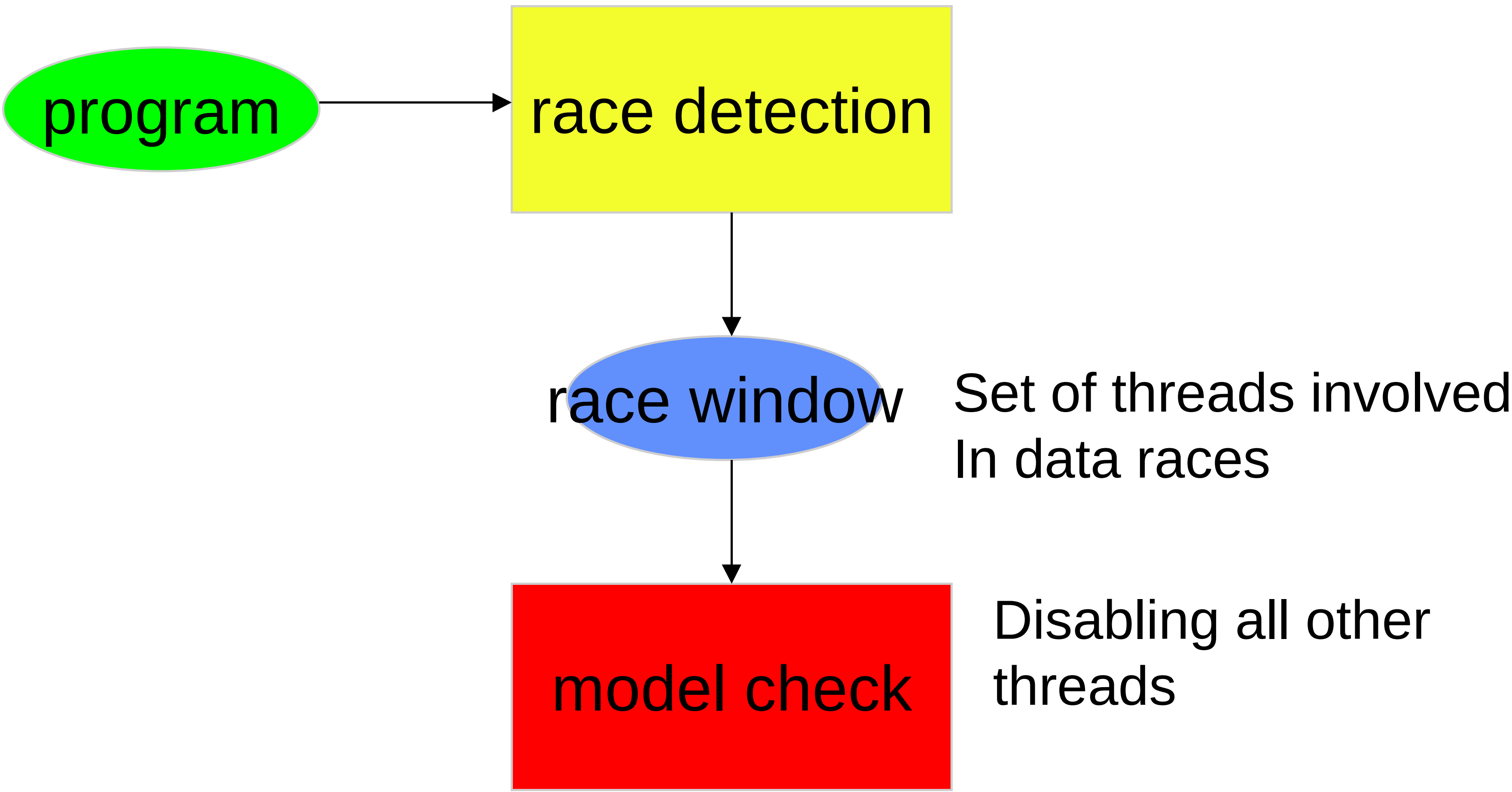
Thread 2:

```
synchronized(L1){
  synchronized(L2){
    synchronized(L3){
      …
    }
  }
};
synchronized(L4){
  synchronized(L3){
    synchronized(L2){
      …
    }
  }
}
```

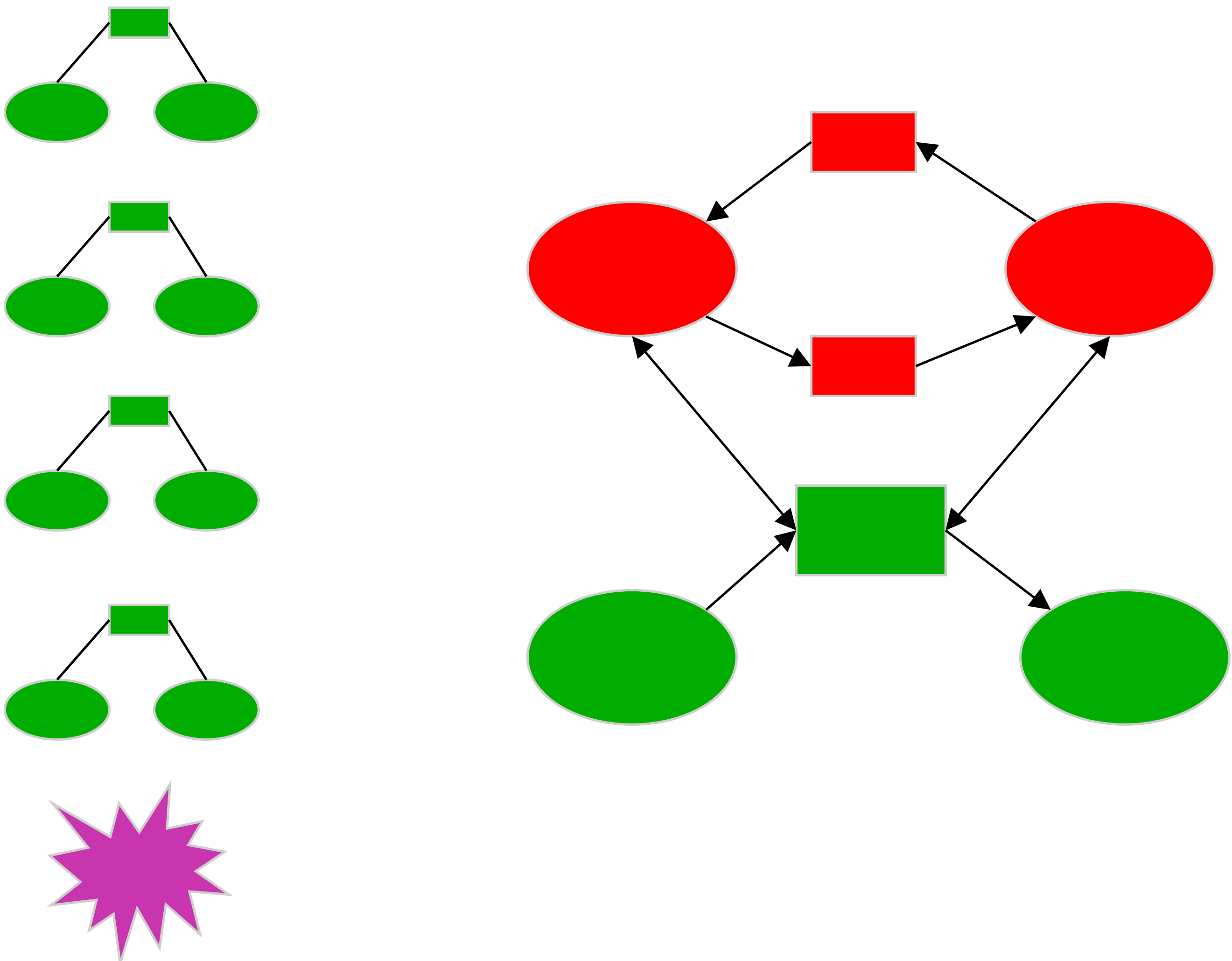
L3 and L4 Are Not Ordered



Integrating With Model Checking



Experiment in State Space Reduction



Conclusions

- Multi-threaded programs require advanced testing approaches. Traditional testing is not satisfactory due to scheduling problem.
- Several NASA specific software applications are multi-threaded. Examples can be found within autonomy and Rover software.
- Model checking and runtime analysis are examples of advanced techniques for testing multi-threaded programs.
- Static analysis, such as extended type checking, can also support testing.
- Future trend: these techniques seem to come together in comprehensive tools. Java PathFinder is one of the first of such tools.
- We collaborate with Kansas State University, Stanford, and Bell-Labs.